THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENT OF ENGINEERING SCIENCE AND MECHANICS

Scientific Figure Captioning with Visual Attention Models

BILL CHEN

SPRING 2022

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Engineering Science
with honors in Engineering Science

Reviewed and approved* by the following:

C. Lee Giles
Professor of Information Sciences and Technology
Thesis Supervisor

Lucas J. Passmore
Professor of Engineering Science and Mechanics
Honors Adviser

*Electronic approvals are on file.

Abstract

Figures are an essential tool for researchers to communicate their complex scientific narratives. However, low-quality captions can cause confusion and misunderstanding among readers which leads to a lost opportunity to convey important research impact. With scientific figures being increasingly important and abundant, automatic figure captioning can enhance the sharing of knowledge between researchers and the community. In this work, we introduced a new dataset based on the Association of Computational Linguistics (ACL) that contains rich scientific figures and captions in large quantity. We analyzed the large-scale dataset and demonstrated the ability of novel attention-based deep neural networks to caption real-world scientific figures. Our experiment results showed the various opportunities and challenges in generating captions for a diverse set of figures.

Contents

List of Figures

List of Tables

Acknowledgement

I would like to thank my research advisors Dr. Lee Giles and Dr. Jian Wu for their outstanding guidance and support throughout my undergraduate research. Their trust and advice allowed me to complete this thesis and their dedication to the development of this field inspired me to become a better researcher. I would like to thank Shaurya Rohatgi for his mentorship and relentless support throughout this and other research projects. I would also like to thank all the members of the CiteSeerX team, for their dedicated work in making the success of various projects possible.

I would like to thank my honors thesis advisor, Dr. Lucas Passmore for guiding me throughout my undergraduate career and helping me achieve my academic goals. I would like to thank the Department of Engineering Science and Mechanics, the Department of Information Science and Technology, and the Pennsylvania State University for their gifts that allowed me to make the most out of my undergraduate education.

Lastly, I would like to thank my family and friends for their support and encouragement throughout college and life.

CHAPTER 1

Introduction

1.1 Motivation

Figures, whether it is a distribution plot or an image of gel electrophoresis, are important to communicate critical research findings and present complicated data. Almost all academic publication today contains a set of figures. Some scholarly search engines and databases have even started to display extracted figures in the thumbnail of publications. Figure captions, being a key component of a figure, is often important to convey complex scientific narratives illustrated by the visual compartments of the figure. Unfortunately, many literature or articles today still have captions that are too generic ("Result of Experiment A") or poorly written ("Relations between X and Y") [25]. Substandard captions can cause confusion and misunderstanding among readers, thus losing the opportunity to convey potentially important research impact.

The use of deep learning to generate image captions has been widely studied in Artificial Intelligence (AI) research. The captioning task requires the machine to detect and understand features within the image and construct a well-formed sentence with semantic language. To achieve this task, methods from both Natural Language Processing (NLP) and Computer Vision (CV) are used in conjunction. Furthermore, the captioning task varies when performed on different image data. Previous work demonstrated novel architectures an techniques such as attention mechanisms that excel in image captioning [54] as well as its use in synthetic scientific figures [7] and extracted figures from arXiv corpus [25]. However, there is still a need for further exploration of extracted large-scale scientific figure data and the application of state-of-art captioning models on those datasets.

1.2 Objectives

The main objective of this project is two-fold. First, we will discuss a newly extracted large-scale scientific dataset from the Association of Computational Linguistics (ACL). From the raw figure dataset, we will extract and analyze the figures and captions independently in order to assess and create subsets for model training. Then, we will train several models based on the attention mechanism and analyze its performance in order to discuss the challenges and opportunities for captioning complex real-world scientific figures.

1.3 Design Needs

There are many components of design needs exhibited in this project. Since the use of natural language processing is critical, we take awareness of the potential biases surrounding language technologies. We followed the ethical guidelines highlighted in [14] where we provide all the details on our machine learning model to offer readers and users transparency. In addition, we also take consideration in the environmental bearings that training large deep learning models can cause [45]. We carefully train and tune our models in order to achieve the same experimental results with minimal drain on the computational resources.

CHAPTER 2

Literature Review

2.1 Machine Learning

This section aims to introduce methods, techniques, and concepts from the field of Machine Learning (ML) that pertain to the topic of this thesis. The reader should be able to follow this chapter and build the high-level knowledge required to understand the task of scientific image captioning. For readers that are already acquainted with topics in deep learning, convolutional neural network, and recurrent neural network, you may skip to Section 2.2.

Machine Learning (ML) is a sub-field of Artificial Intelligence (AI) where computer algorithms improve automatically through and by the use of data [36]. The application of ML can be frequently found in predictive tasks, such as classifying hand-written digits with a Support Vector Machine [40]. Additionally, practitioners also use unsupervised ML methods such as k-Means clustering for exploring complex data sets [30].

2.1.1 Deep Learning

With the increasing demand for addressing more complex tasks, deep learning (DL) was developed as a sub-field of ML. Deep learning methods are based on artificial neural networks (ANNs) inspired by information and distributed communication nodes in biological neurons [35]. Those biological neurons produce short electrical impulses which travel along the axons and make the synapses release chemical signals (information). When another neuron receives a sufficient amount of those signals, it fires its own electrical impulses. When those simple behaving neurons are organized in a network of billions, it enables highly complex computations [20]. Figure 2.1 represents the structure of a biological neuron.

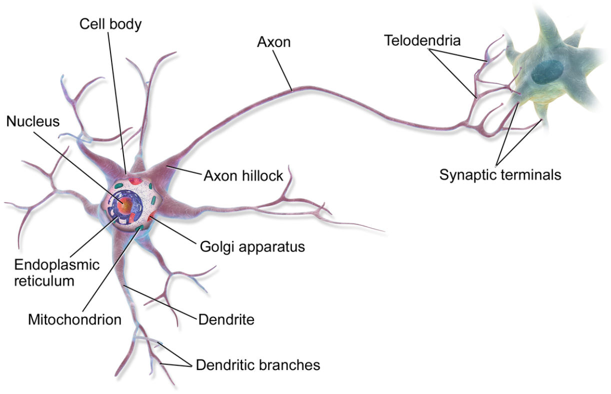


Figure 2.1: Biological Neuron

The simplest architecture of ANN is the perceptron model presented in Figure 2.2. The inputs $x_0, x_1, \ldots, x_n$ can be represented with $\boldsymbol{X}$. The weights $w_0, w_1, \ldots, w_n$ can be represented with $\boldsymbol{W}$, and the bias $b_0, \ldots, b_n$ (not shown in figure) can be represented as $\boldsymbol{b}$. With linear algebra, you can efficiently compute the output O from this perceptron of n inputs with Equation 2.1 where ϕ is the activation function [20]. The activation function introduces non-linearity to the model. A common activation function in practice is the Rectified Linear Unit (ReLU) [22].

$$O_{W,b}(X) = \phi(XW + b) \tag{2.1}$$

$$(ReLU)\ \phi = f(x) = x^{+} = max(0, x) \tag{2.2}$$

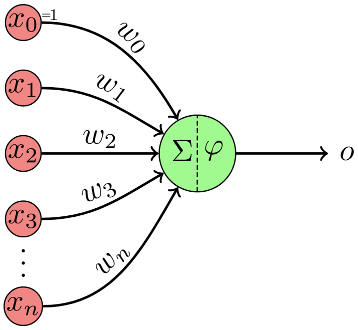


Figure 2.2: Perceptron Model [13]

Now, this simple perceptron model can be expanded by forming multiple layers of fully connected neurons. The multilayer perceptron is consists of an input layer, one or more hidden layers, and an output layer (Figure 2.3). The hidden layers accept outputs from previous layers as their inputs. When an ANN contains a deep stack of hidden layers, it is called a deep neural network, thus the study of it being named deep learning [20].

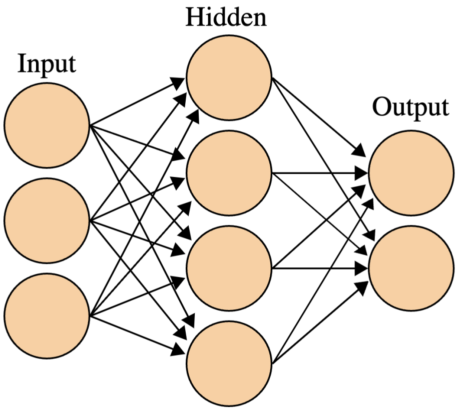


Figure 2.3: Multilayer Perceptron [11]

In the subsequent sections, we will see how the classic structure of ANN is modified to form new architectures that adapt to more complex and specific tasks such as computer vision and natural language processing.

2.1.2 Convolutional Neural Network

A class of artificial neural network that sees wide application in computer vision tasks is convolutional neural network (CNN). As a regularized version of multilayer perceptrons, CNNs also draw inspiration from biological systems. Specifically, CNN resemble the process used by animal visual cortex, where individual cortical neurons respond to signal in a restricted region of the visual field [19].

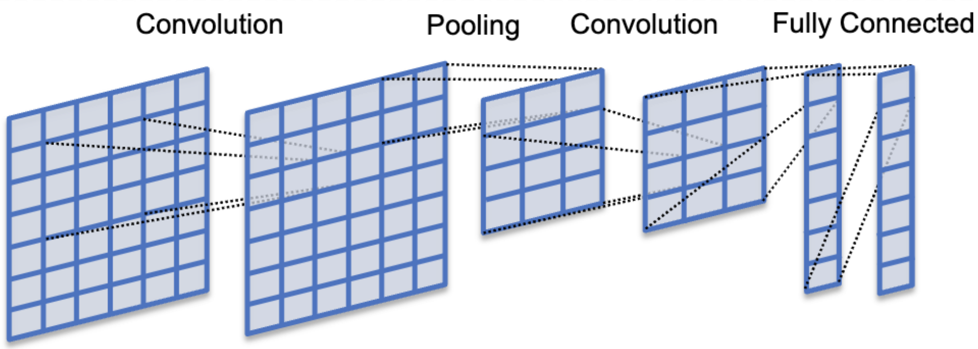


Figure 2.4: Typical Convolutional Neural Network [12]

Like MLP, CNN is constructed with multiple layers and has an input layer and an output layer. However, the convolutional layers in between the input and output layers gives CNN its power. A typical convolutional layer has three stages. In the first stage, named convolution stage, a dot product is computed between a sub-field of the input matrix with a same size "filter" matrix named the kernel. The result of the computation replaces the original input field matrix with its output. This operation is repeated by shifting through the entire input matrix, effectively convoluting the input. The second stage, named detector stage, runs the linear activation outputs from the first stage through a nonlinear activation function. In the third stage, named pooling stage, a pooling function replaces the output at certain location with summary statistics of nearby outputs [22]. One example of this is max pooling, where the operation returns the maximum output within a rectangular neighborhood [57].

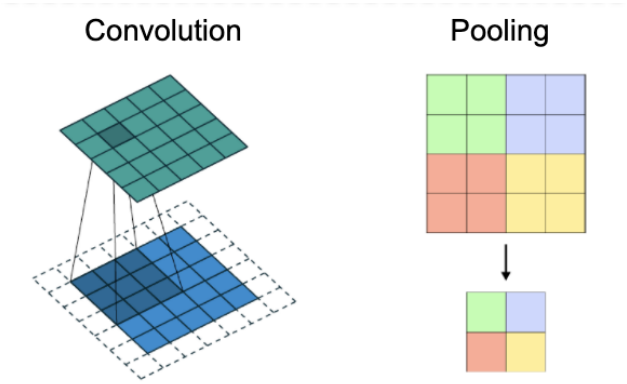


Figure 2.5: Convolution and Pooling Visualization [12]

In image captioning tasks, CNNs are often used for feature learning or feature extraction. Different CNN architectures can be applied to extract local or global features, depending on the goal of extraction. Typically, the features are then fed to a Recurrent Neural Network (RNN) for caption generation.

2.1.3 Recurrent Neural Network

A shortcoming of traditional feed-forward neural network is that it assumes input as independent data points. This means that it will have issues working with sequence data where one data point depends on previous data point. Much like how CNN specializes in processing grid of values such as an image, another class of ANNs called the Recurrent Neural Networks (RNNs) specializes in processing sequence of values $x^{(1)}, \dots, x^{(n)}$. Some examples of those sequences includes text and speech [22].

The simplest example of recurrence can be illustrated with a single recurrent neuron. The neuron will receive an input x and produce an output y, much like a regular neuron in a feed-forward neural network. The difference comes when the neuron sends its output back to itself [20].

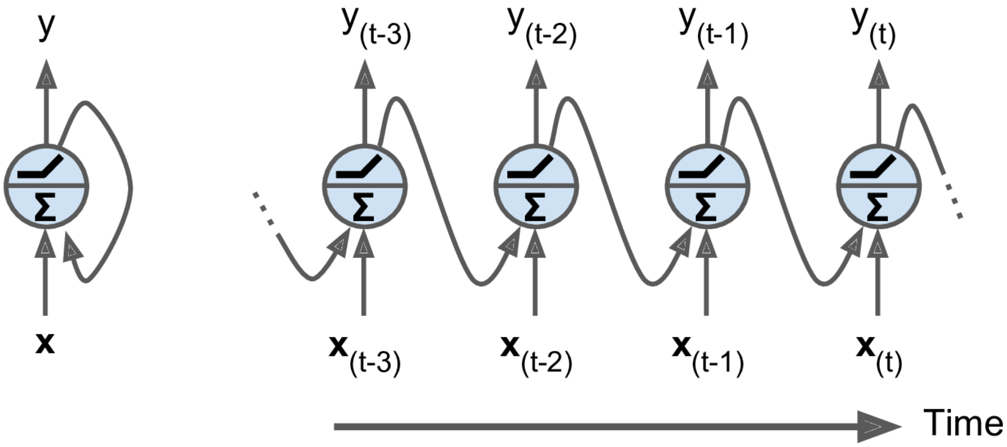


Figure 2.6: Recurrent neuron (left) unrolled through time (right) [20]

A layer of recurrent neurons can be easily created (Figure 2.7). At each time step t, the recurrent layer would now take both the input vector $\boldsymbol{x}_{(t)}$ and the output vector from previous time step $\boldsymbol{y}_{(t-1)}$. Each recurrent neuron has two sets of weights $\boldsymbol{W}_x$ and $\boldsymbol{W}_y$ that are weighted matrices, a bias vector $\boldsymbol{b}$, and a activation function ϕ. The output of a recurrent layer can be computed with the following equation [20]:

$$\boldsymbol{y}_{(t)} = \phi\left(\boldsymbol{W}_x^T \boldsymbol{x}_{(t)} + \boldsymbol{W}_y^T \boldsymbol{y}_{(t-1)} + \boldsymbol{b}\right) \tag{2.3}$$

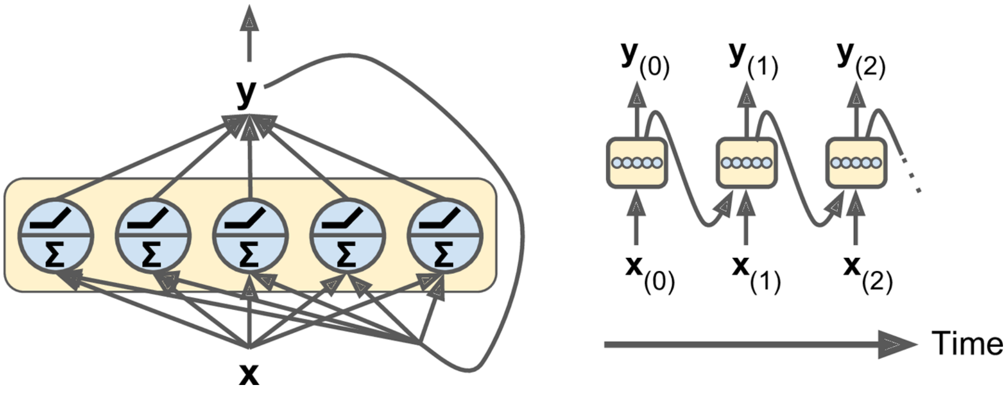


Figure 2.7: A layer of recurrent neurons [20]

Since the output of a recurrent neuron contains information from previous time steps, it produces a form of *memory*. This is why RNNs are frequently used for sequence data where dependence is important. Although the basic memory cell is sufficient to illustrate the concept of RNN, it suffers from the vanishing gradient problem. When the pattern in question is too long, information will be lost as it traverse through the network [4]. In 1997, the Long Short-Term Memory (LSTM) cell was introduced by Sepp Hochreiter and Jürgen Schmidhuber [23]. Those LSTM cells are widely used in complex natural language processing tasks such as caption generation as we will see in the next section [24].

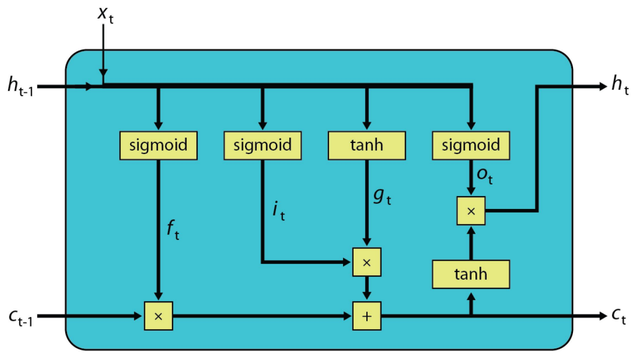


Figure 2.8: Architecture of a LSTM cell [21]

Figure 2.8 shows the internal operations of a LSTM cell. The idea behind LSTM cells is to have the network learn what to store in the long-term state and when to forget old states. The cell has the same input and output as a regular recurrent unit (input at time step t $\boldsymbol{x}_{(t)}$ and function of some inputs at that time step t and its state at the previous time step $\boldsymbol{h}_{(t)}$). But there is an additional system to control the flow of information. The most important component is $x^{(t)}$ which performs a self-loop operation. The self-loop weight is controlled by the **forget gate** unit $f_i^{(t)}$ [22]:

$$f_i^{(t)} = \sigma\left(b_i^f + \sum_j U_{i,j}^f x_j^{(t)} + \sum_j W_{i,j}^f h_j^{(t-1)}\right) \tag{2.4}$$

The internal state is then updated with the following equation:

$$x^{(t)} = f_i^{(t)} x^{(t-1)} + g_i^{(t)} \sigma\left(b_i + \sum_j U_{i,j} x_j^{(t)} + \sum_j W_{i,j} h_j^{(t-1)}\right) \tag{2.5}$$

The **external input gate** $g_i^{(t)}$ in Equation 2.5 is calculated with similar equation to the forget gate, except with it's own parameters. Lastly, the output $h_i^{(t)}$ can also be shut off via the **output gate** $o_i^{(t)}$:

$$h_i^{(t)} = tanh\left(s_i^{(t)}\right) q_i^{(t)} \tag{2.6}$$

$$o_i^{(t)} = \sigma\left(b_i + \sum_j U_{i,j} x_j^{(t)} + \sum_j W_{i,j} h_j^{(t-1)}\right) \tag{2.7}$$

There is another popular recurrent unit model that serves similar purpose as LSTM named the Gated Recurrent Unit (GRU) [9]. The architecture of GRU cells are a simplified version of LSTM cells but showed similar performance. For more detail on GRU we refer the user to the following notes [22] [20].

2.2 Image Captioning Models

Given the importance of image captioning in everyday applications, it has been a popular field of research in the past few years. The earlier methods proposed for captioning are mostly template based. In those approaches, fix templates with blank slots are filled with extracted features from the image space. For example, Farhadi et al. [17] uses a Markov Random Field classifier to extract a triplet representation <*object, action, scene*> from the image and sentence space to generate a final description by calculating similarity. Li et al. [31] developed further on that idea by composing sentence automatically from scratch with web-scale n-grams. In a similar fashion, Kulkarni et al. [29] uses a Conditional Random Field to infer a triplet representation consisting of objects, attributes, and prepositions to generate descriptions. Although the above methods provided a way to generate grammatically correct descriptions, they do have several shortcomings. Due to the nature of template based methods, the caption outcome is fixed in length and by many specific rules. Those hinder the model's ability to generate variable and consistent captions. Recent developments in novel deep learning methods for captioning has given us a better way to generate natural language that mirror the creativity and attention of a human doing such task.

The general approach for the deep learning methods involves extracting image feature with a CNN based encoder, and then describing those features through a RNN based language model [24][53]. This is essentially the encoder-decoder architecture that we will discuss in the following section. However, other methods that includes a composition of additional networks and the use of attention mechanisms has been added to this basic structure for better performance.

2.2.1 Encoder-Decoder Architecture

The encoder-decoder technique has been popular and effective in areas such as neural machine translation [46]. This has inspired its use for image captioning. The encoder-decoder based methods generally involves the following steps:

1. Extract image features with the hidden activation of CNN
2. Feed the extracted feature to a language model for converting to words.

The process is illustrated in Figure 2.9

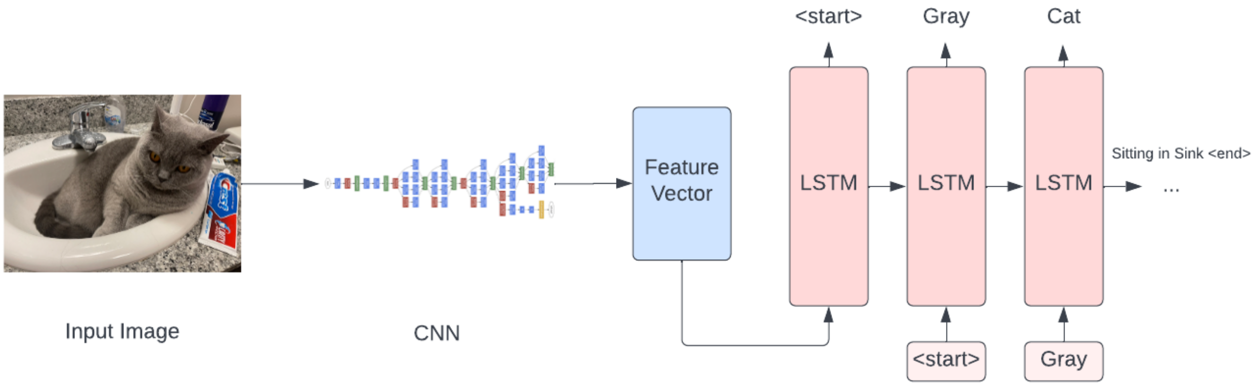


Figure 2.9: Basic Encoder-Decoder based Image Captioning Process

Although the general model design is similar, researchers have experimented with different variations of CNN, language models and training methods. For instance, Vinyals et al. [52] proposed a method called Neural Image Caption Generator. This method uses a CNN with novel batch normalization that employ its output from the last hidden layer as input to a LSTM decoder. Because the input image is fed once only at the beginning of the process, even the LSTM suffer from the vanishing gradient problem when trying to generate long captions. Jia et al. [26] proposed a extended LSTM called guided LSTM (gLSTM) which includes global semantic information to each gate and cell state in order to combat this problem.

2.2.2 Compositional Architecture

Similar to the encoder-decoder architecture, the compositional architecture includes a CNN component for image understanding and a RNN component for text generation. However, compositional methods treat those components and several other functional components as separate blocks. A general compositional architecture based method includes the following steps [24]:

1. Extract image feature with CNN
2. Obtain visual concepts and attributes from visual features
3. Generate multiple captions with language model using the output of Step 1 and Step 2
4. Use a deep multi-modal similarity model to select high quality captions

The process is illustrated in Figure 2.10

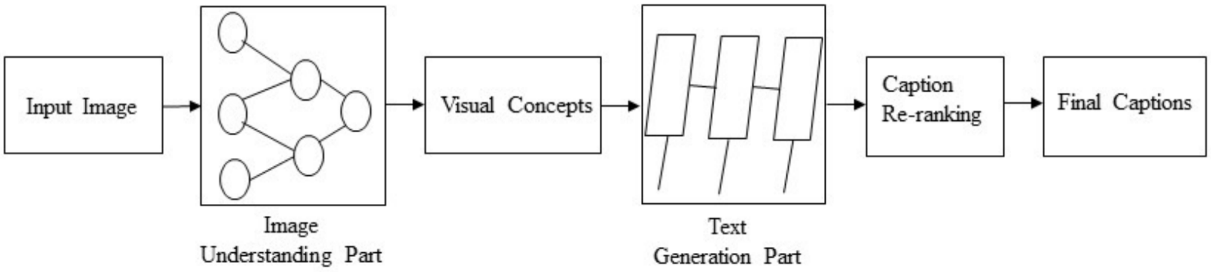


Figure 2.10: Basic Compositional Architecture based Image Captioning Process [24]

Fang et al. [16] introduced a system that uses a visual detector in image sub-regions rather than full images. A maximum entropy language model was used to generate image captions from those sub-region features. The system then re-rank generated captions by linear weighting of sentence features. Researchers also use a variety of additional function blocks for specific captioning applications. Most methods mentioned earlier perform well for captioning tasks on the same domain, but there is no certainty that those methods perform well on open-domain images. Tran et al. [48] introduced a compositional based model that uses an external knowledge base to recognize entities such as landmarks and celebrities. This methods has proven effective even in open-domain databases where key entities are not in the model training scope.

2.2.3 Attention Based Methods

Most methods we saw earlier were able to generate caption for an input image through a feed-forward fashion. However, a common issue with those methods is that they aren't able to analyze the image over time as they generate captions. Furthermore, they do not consider spatial aspects of the image relevant to parts of the image caption. Attention mechanisms have been popular in deep learning to address those limitations by dynamically focus on parts of the input while outputs are being produced. Xu et al. [54] first introduced a method leveraging attention mechanisms for image captioning. Rather than compressing an entire image into a static representation, the method use attention to bring salient features to the forefront as captions are generated. The paper proposed two different attention mechanisms. The "soft" attention mechanism focus on all regions and provide weight value to each region while the "hard" attention mechanism select focused areas based on probability value (Figure 2.11). An additional benefit to this method was that we can "see" how the model maps its attention region to specific output text. This will be critical for exploring how attention based models caption scientific figures differently from other images.

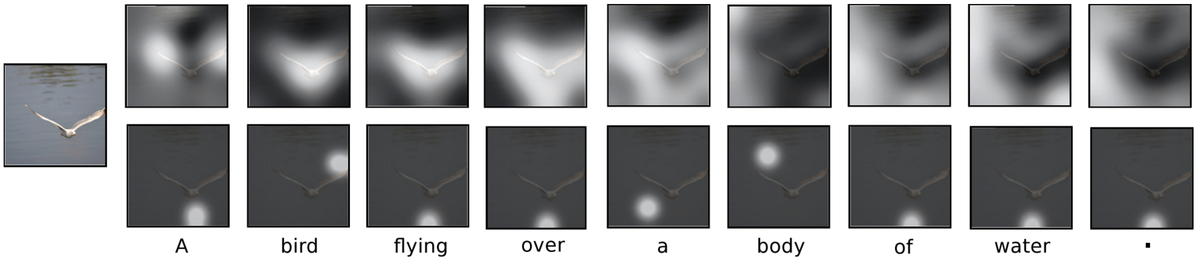


Figure 2.11: Demonstration of Attention Based Image Captioning with "hard" attention mechanism (bottom row) and "soft" attention mechanism (top row). [54]

Generally, a attention based method includes the following steps [24]:

1. Obtain image information based on the whole scene with a CNN
2. Generate words and phases based on the output of Step 1 with a language model
3. Focus on salient regions of the image with each time step of the language generation
4. Update captions dynamically until the end state of language model

A diagram of the process is provided in Figure 2.12. Many other attention based methods have been introduced in literature. For example, Jin et al. [27] proposed a method that was able to extract flow of abstract meaning based on semantic relationship between visual and textual information. Yang et al. [55] introduced a review-based attention method that perform multiple review steps with attention on CNN hidden states. Due to its high performance in generating meaningful captions, similarity with human visual processing functions, and ability to inspect caption stages, attention based mechanisms have been popular in the field of image captioning [38][33][8].

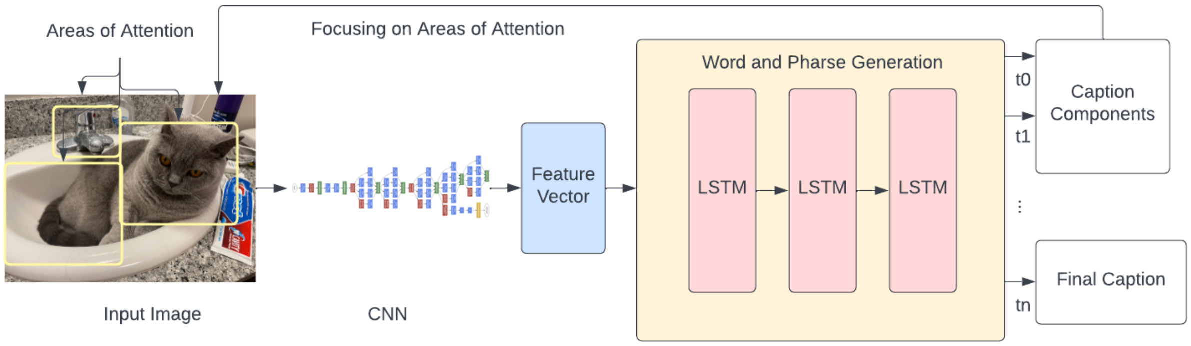


Figure 2.12: Basic Attention Mechanism based Image Captioning Process

2.2.4 Evaluation Metrics

Evaluation metrics are necessary for assessing the performance of any machine learning model. In simple classification models, metrics such as accuracy, precision, recall, and F-1 score are used to evaluate its performance [56]. However, more sophisticated metrics have been developed or adopted to evaluate models for image captioning:

BLEU [37]
Bilingual Evaluation Understudy (BLEU) is the most popular metric used to measure the quality of machine generated text. The score calculates the ratio of prevision value for n-gram between generated phrases and reference phrases. The score also implements penalty mechanisms to address the problem of short prediction sequence and geometric average to balance the difference of precision values of n-gram. Although a generally reliable metric, there are limitations to BLEU scores when the generated text is short [6].

ROGUE [32]
Recall-Oriented Understudy for Gisting Evaluation (ROGUE) is a set of metrics for measuring the quality of text summary. The method compares word sequences, word pairs, and n-grams with a set of human curated reference summaries. In particular, ROGUEL, ROGUEW, and ROGUES are metrics included in the set that are appropriate for evaluating figure captioning.

METEOR [3]
Metric for Evaluation of Translation with Explicit ORdering (METEOR) is another metric used for machine translated language. In this method, standard word segments are compared with reference texts and stems of sentences. In addition, synonyms of words are also considered for matching. METEOR exceeds at comparing sentence level correlations.

CIDEr [51]
Consensus-based Image Description Evaluation (CIDEr) is a consensus metric for evaluating image descriptions. CIDEr achieves human consensus by using term frequency-inverse document frequency (TF-IDF).

SPICE [1]
Semantic Propositional Image Caption Evaluation (SPICE) is a newer metric for evaluating quality of image captions. This method uses a graph-based semantic representation to extract information of different objects, attributes, and their relationship to the image description.

2.2.5 Scientific Figure Captioning

In this section, we will review related works that are specific to the captioning of scientific figures. Chen et al. [7] identified some challenges when captioning figures as compared to regular images. Those challenges revolve around "pivot" elements in the object. For example, in the image with the cat in a sink (Figure 2.12), the "pivot" elements are the cat and the sink. However, in a bar graph, all the bars, axis, and labels can be pivot elements. Additionally, the model also has to determine how important a "pivot" element is compared to other "pivot" elements. In the same work, Chen et al. introduced FigCAP, a figure-caption pair corpus with synthesized figures and five models to perform captioning on the dataset. Some of the models experimented include a CNN-LSTM baseline, CNN with LSTM plus multiple attention assisted decoders, and a combine model with CNN-LSTM with attention mechanisms and reinforcement training. They found that the models with attention metrics outperforms the baseline CNN-LSTM model.

Hsu el al. [25] identified the problem where model trained on synthesized figures and captions tend to only describe features without any other high-level insights. They addressed this issue by developing SciCAP, a figure-caption pair corpus based on the arXiv dataset which contains extracted figures and captions from arXiv documents [10]. Furthermore, they implemented a baseline CNN-LSTM model with an attention mechanism on the graph plots sampled from the dataset. Their results show the performance of the captioning model is the highest on figures with single-sentence captions. Though, there are general challenges when captioning real-world figures since the baseline model has a unsatisfactory BLEU-4 score and empirical results.

CHAPTER 3

Data Processing and Analysis

3.1 Dataset

In the previous chapter, we discussed two figure-caption paired datasets: FigCAP [7] which contains synthesized figure-caption pair data, and SciCAP [25] which contains extracted figure-caption corpus from arXiv. In this thesis, we increase the quantity and diversity of scientific figures by constructing and utilizing a dataset extracted from the Association for Computational Linguistics (ACL) corpus.

3.1.1 Extraction Pipeline

The ACL document corpus contains 55,760 research papers. The goal of the pipeline is to extract figures and tables from those research papers. The extracted data is then augmented by automatically extracting reference context and linking them to corresponding figures. For figure extraction, we used DeepFigure [42], a distant supervised learning method to induce labels of figures from a large collection of scientific documents. The extraction pipeline is summarized in the following steps:

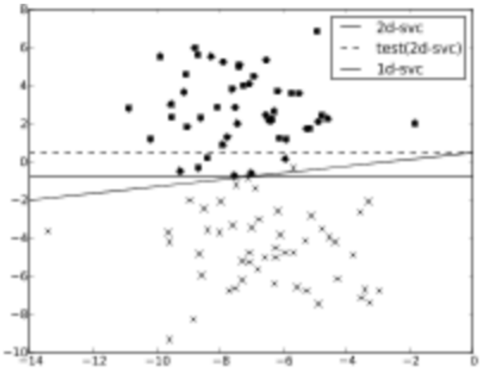
1. Retrieve identifier from the job queue and pull paper from the server file system

2. Run DeepFigures on the paper to detect figures and extract captions

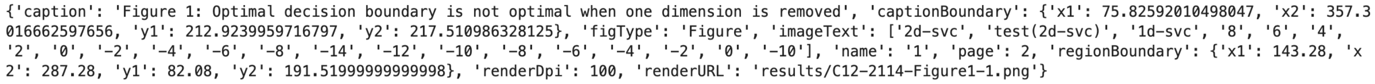
3. Crop the figures from the rendered PDFs

4. Save the cropped figures as .png files in the resulting directory

5. Save the metadata file in JSON format into a directory identified by PDF file name



(a) extracted cropped figure sample



```
{'caption': 'Figure 1: Optimal decision boundary is not optimal when one dimension is removed', 'captionBoundary': {'x1': 75.82592010498047, 'x2': 357.3
016662597656, 'y1': 212.9239959716797, 'y2': 217.510986328125}, 'figType': 'Figure', 'imageText': ['2d-svc', 'test(2d-svc)', '1d-svc', '8', '6', '4',
'2', '0', '-2', '-4', '-6', '-8', '-14', '-12', '-10', '-8', '-6', '-4', '-2', '0', '-10'], 'name': '1', 'page': 2, 'regionBoundary': {'x1': 143.28, 'x
2': 287.28, 'y1': 82.08, 'y2': 191.51999999999998}, 'renderDpi': 100, 'renderURL': 'results/C12-2114-Figure1-1.png'}
```

(b) extracted figure metadata sample

Figure 3.1: Example of extracted figure-metadata pair from ACL papers

After applying the extraction pipeline to the corpus of 55,760 ACL papers, we were able to extract a total of 264,343 individual figures and tables. Those figures and tables are stored as individual .png files under the result directory along with a .json file, consisting of a list of metadata for figures and tables in a single paper. An example of the figure-metadata pair is shown in Figure 3.1.

3.1.2 Data Processing

From the sample data, we observe that extensive data processing is required to extract the figure-caption pair and reformat it for model training purposes. In the following sections, we will go into detail about the methods and procedures used to preprocess figure and caption data that will enable analysis and captioning experiments. This will involve extracting the figures and captions from the directory that contains the total information. Then, we will apply independent data cleaning procedures to the figure and caption data. Finally, we will perform an analysis of the figures and captions and combine them into a new dataset appropriate for our captioning experiments. Figure 3.2 provides a high-level illustration of the entire process.

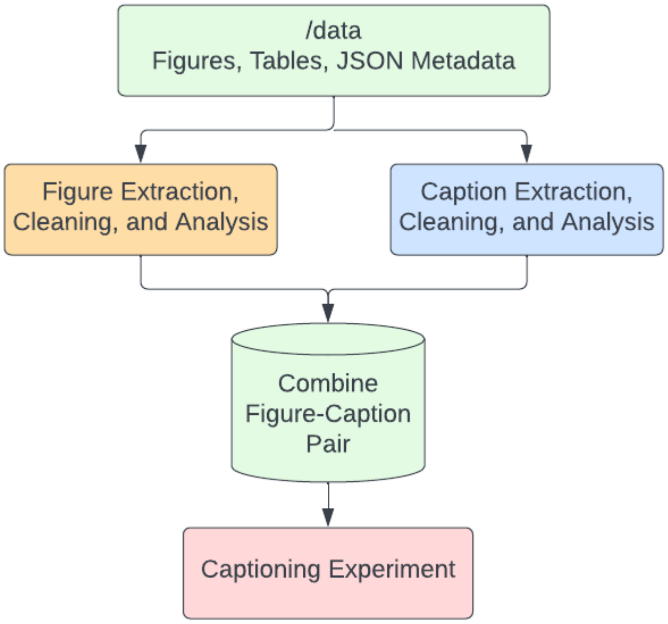

/data
Figures, Tables, JSON Metadata
Figure Extraction, Cleaning, and Analysis
Caption Extraction, Cleaning, and Analysis
Combine Figure-Caption Pair
Captioning Experiment

Figure 3.2: High-level data processing procedure

3.2 Figure Processing and Analysis

3.2.1 Processing

The high-level figure processing steps are illustrated in Figure 3.3.

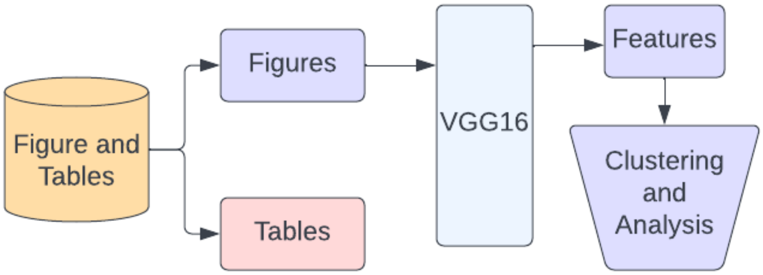


Figure 3.3: High-level figure processing steps

Since tables contain an entirely different feature structure than other scientific figures, our first step would be to remove them from our dataset (298,856 images). In the directory, the figures and tables are indicated in the file name during our extraction step. We use a regular expression [47] to separate out the figure files. After the separation, we have 123,721 figures and 175,135 tables. Only the figures will now move to subsequent processing.

In order to use the figures for analysis and model training, we will digitize them by extracting features with a CNN-based architecture. The VGG16 model [43] is a large series of convolutional layers followed by fully connected layers. It is suited for this task because of its state-of-art performance when coming to similar tasks [44]. We load the VGG16 model with the pre-trained weights from the ImageNet dataset [39]. Then, we removed the output layer so the final output will be a vector of image features. Before putting the image through the models, we reshape them to 224x224x3 in order to match the pre-defined input size of VGG16. The result for feature extraction was 123,721 (number of images) samples with a feature vector length of 4096 (defined feature size from hidden layer).

To increase computational efficiency when analyzing the figure data, we will need to reduce the dimension of our feature vector. Principle Component Analysis (PCA) [18] is a standard statistical procedure to perform dimensionality reduction. The process generates hyper-planes in n-dimension captured by principle components that retain variance within the data. Before running PCA on our feature vectors, we need to determine the final number of principle components to preserve. We do this by fitting PCA on our feature vectors and calculating the cumulative explained variance at the different count of principle components (Figure 3.4).

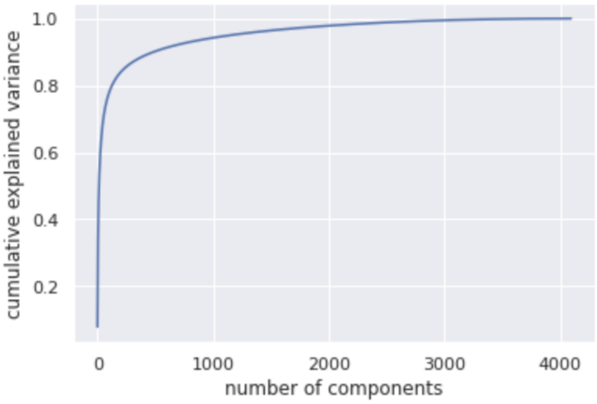


Figure 3.4: Cumulative explained variance of different principle component count

Typically, we aim to retain at least 80% of the cumulative explained variance when selecting the number of principle components. From Figure 3.4, we see that with 500 components, we can retain 85% of the variance. This choice gives us the greatest boost in computational efficiency while retaining important features of each figure image.

3.2.2 Analysis

Since the dataset is extracted from a large corpus of real scientific literature, we know that there will be many types of figures. Our main objective with figure analysis is to observe and understand how to group and classify the figures in our dataset. This is a necessary step in order to both understand the dataset and interpret experimental outcomes for the captioning model.

To group the figures, we use K-Means clustering [34], an unsupervised method for partitioning data into k clusters. A metric to measure the separation of clusters is the Silhouette score. The score measures how similar an object is to its own cluster compared to other clusters. A higher Silhouette score indicates that the object is highly cohesive with the assigned cluster. We run the K-means clustering algorithm over the feature vectors where we set the number of clusters **k** from 2 to 10. For each **k**, we calculate the respective Silhouette score (Figure 3.5).

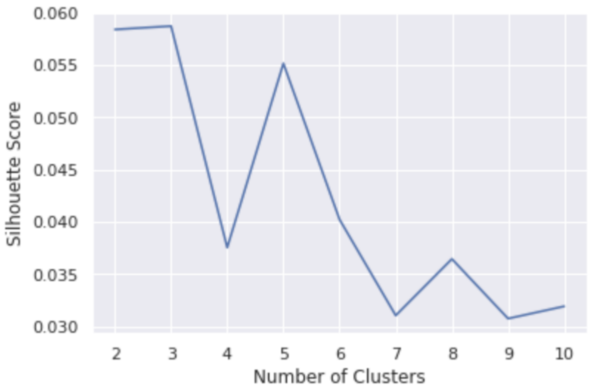


Figure 3.5: Silhouette score for different numbers of figure clusters

The Silhouette score has peaks at $k = 3$ and $k = 5$. To increase diversity, we assigned each figure to a cluster based on the $k = 5$ results. We visualized the clusters by performing another PCA operation, this time reducing the data to 2 principle components. The results are shown in Figure 3.6. In addition, we performed t-Distribution Stochastic Neighbor Embedding (t-SNE) [50] (Figure 3.7), another unsupervised clustering technique that differs from PCA by preserving small pairwise distances and local similarities between data points. Due to their differences, t-SNE can provide extra information in addition to PCA when visualizing large-scale datasets.

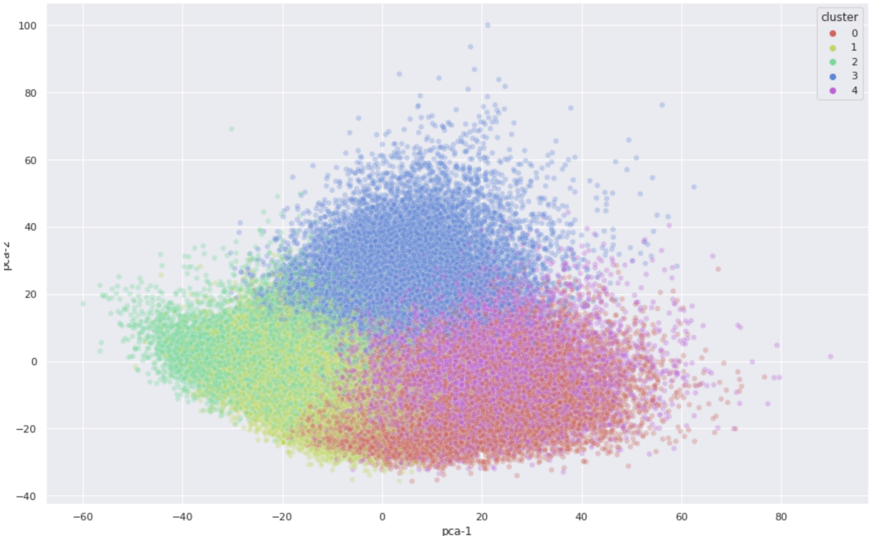


Figure 3.6: 2D visualization of figure clusters

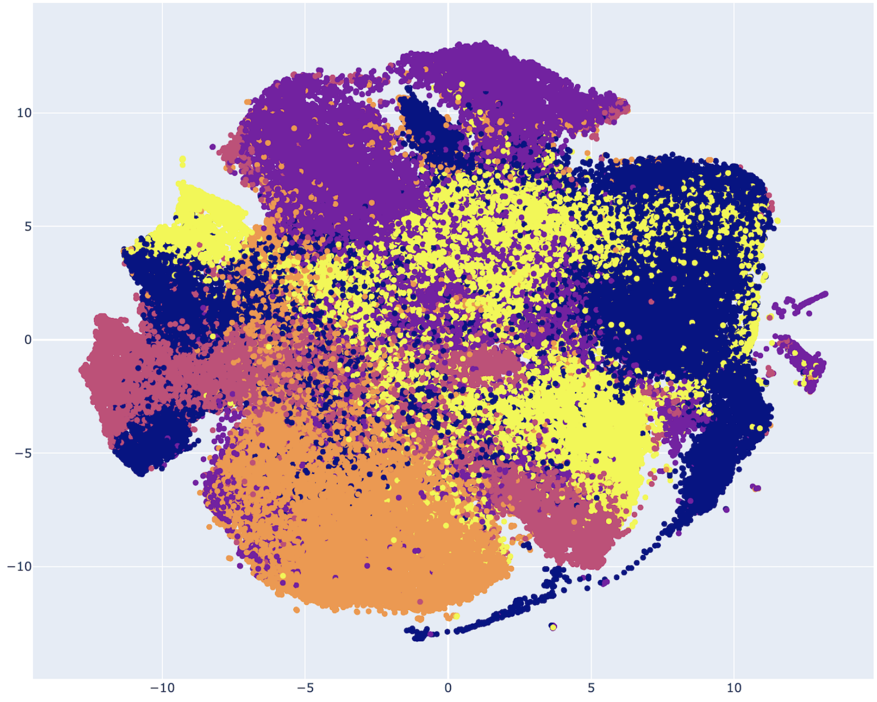


Figure 3.7: Visualization of figure clusters with t-SNE

Although we separated the figures into 5 clusters, we observe that there are loosely 3 clusters in Figure 3.6. Clusters 1, 2, and clusters 0, 4 exhibit high degree of overlapping. This is reflected in the high Silhouette scores for 3 clusters. The t-SNE visualization provided similar information where major areas of the plot is overlapped by figures from different clusters. The results suggested that many figures contain features that are highly similar. This could be expected since the two types of figures share similar "pivot" features. For example, flowcharts that contain a heavy amount of text might be clustered with a manuscript figure. Although they are classified as different types of figures, the appearance of textual information can cause them to be assigned the same cluster.

In Table 3.1, we selected representative sample figures from each cluster. We grouped clusters 0, 4, and clusters 1, 2 since the majority of figure types in those clusters are shown to be similar. By observing and labeling sampled figures closest to the centroid of each cluster, we provide an empirical label for the cluster. Clusters 0 and 4 contain figures with text-heavy features such as flowcharts and screenshots of text documents. Cluster 3 contains line graphs and scatter plots. Clusters 1 and 2 contain mostly graphs with block-like features such as bar graphs. However, some figures in clusters 1 and 2 also scatter plots or line graphs that are more cluttered, creating a block-like view. The same is true for cluster 3. The results suggested the complexity of analyzing distinct features for automatically extracted scientific figures. Distinguishing functionalities between different figure types purely based on visual features extracted by a CNN is shown to be difficult. It also suggests that the performance of the captioning model, which also uses extracted visual features, can be affected due to unclear "pivot" elements.

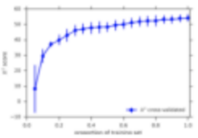
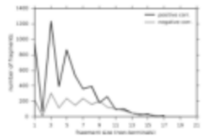
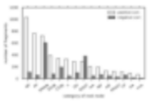
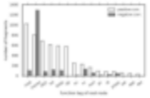
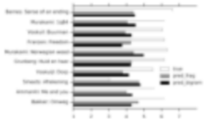
Cluster	Sample Figures	General Figure Types
Clusters 0 and 4		Text-heavy Flowcharts and Images
Cluster 3	 	Scatter Plots and Line Graphs
Clusters 1 and 2	  	Bar Graphs and Histograms

Table 3.1: Sample figures from clusters with empirical classification

3.3 Caption Processing and Analysis

3.3.1 Processing

The high-level caption processing steps are illustrated in Figure 3.8.

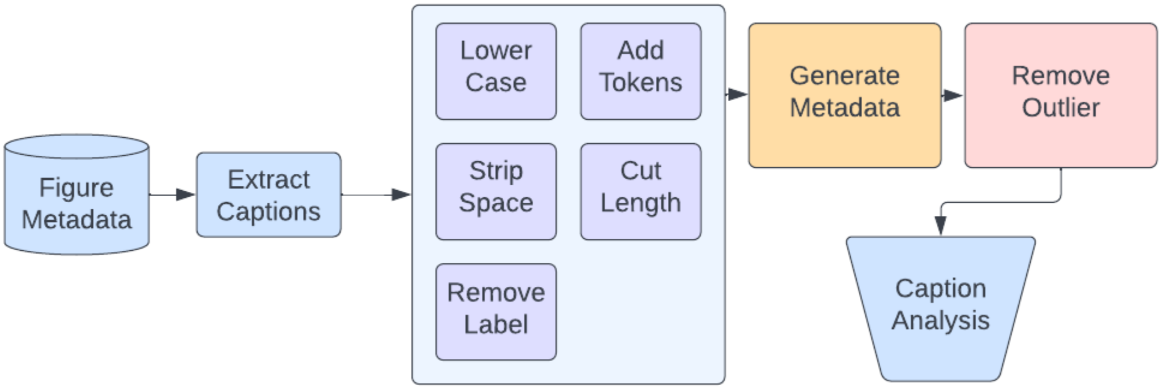


Figure 3.8: High-level caption processing steps

The caption is currently stored as a key in the figure metadata in JSON format. The first step involves extracting the captions from each metadata file. Since we excluded tables for captioning, we will need to exclude all captions belonging to a table. Since most of the captions contain a label (ex: Figure 2-3 shows the flow of syntactic and ...), we can use a regular expression to match only captions that belong to a figure.

After the features are extracted, we apply a series of text operations to format it for model training. We first standardize the text by lower the casing and removing extra white space. Then, we removed the labels (Figure 2-3, Table 1-1, etc.) since they will be trivial for learning. An important step here is to add tokens <start> and <end> to the start and end of the captions. Those tokens will signal the model to start and end sequence generation.

With the caption text cleaned and formatted, we can now generate some metadata such as the number of words and number of sentences. The Natural Language Toolkit (NLTK) [5] provides the functions to compute those metadata. We then move on to removing outliers based on the number of words and number of sentences. We computed the interquartile range (**IQR**) and set cutoff limits with:

$$upperlimit = 75\%percentile + 1.5 \times IQR \tag{3.1}$$

$$lowerlimit = 25\% percentile - 1.5 \times IQR \tag{3.2}$$

The process removed 8775 outliers from the dataset. The distribution of caption word count before the removal process is displayed in Figure 3.9.

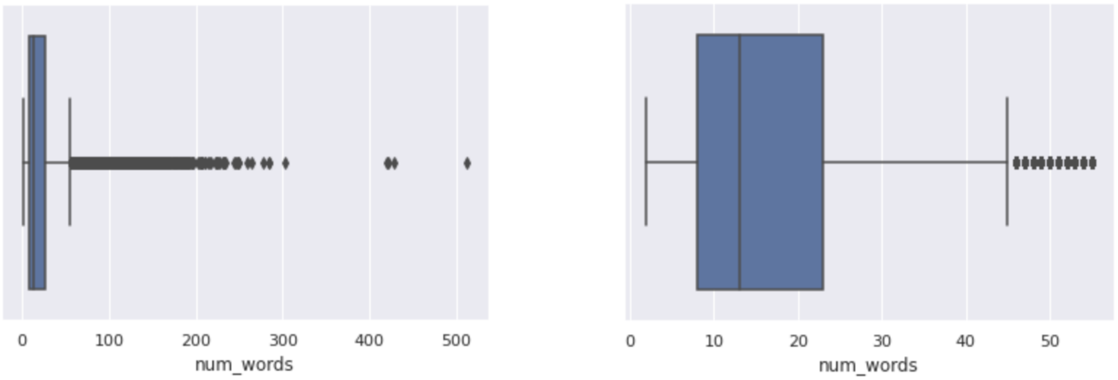
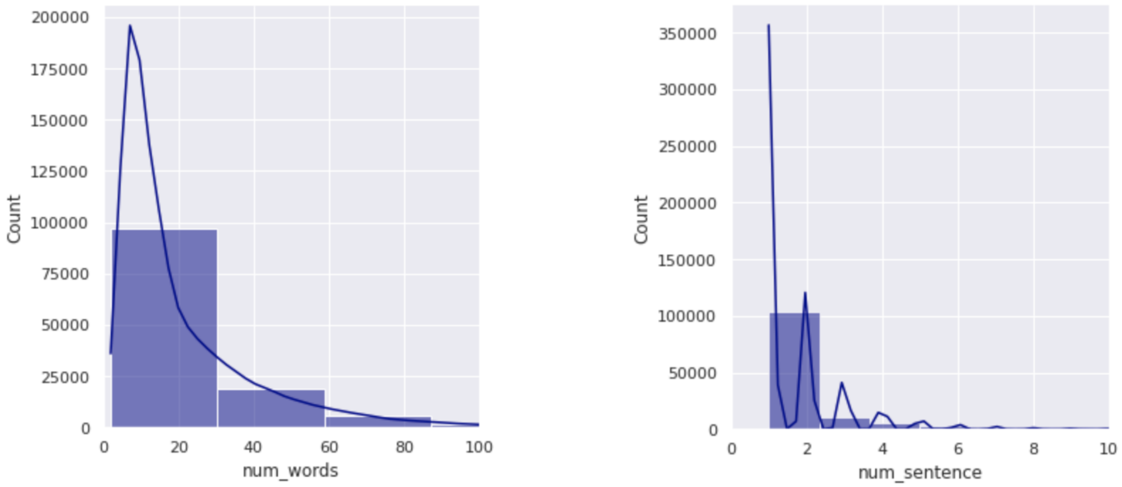


Figure 3.9: (a) Distribution of word count before removing outliers and (b) Distribution of word count after removing outliers

Based on the results in [25], we see that captioning often suffers in performance when the captions get long. Therefore, we only take the first 3 sentences of any captions that are longer than 3 sentences. Lastly, we know previously from our figure analysis that the extraction results contain many figures with sub-figures. In order to remove those, we filter all the captions that contain subsection labeling (ex: (a)...(b)...(c)...). 3955 total captions are identified for indicating sub-figures and have been removed from the final caption dataset.

3.3.2 Analysis

The automatic extraction process of captions from real-world scientific literature likely means that the caption qualities will vary. With the standardization process in place, the number of words and sentences in a caption becomes the most important caption quality that can affect caption generation. We can visualize the overall distribution of word count and sentence count with distribution plots (Figure 3.10). The plots indicate that most captions in the dataset are below 30 words and 2 sentences. However, there are a decent amount of captions that have 40 to 60 words and some examples that can be over 100 words. Similarly, some captions will have more than 10 sentences.

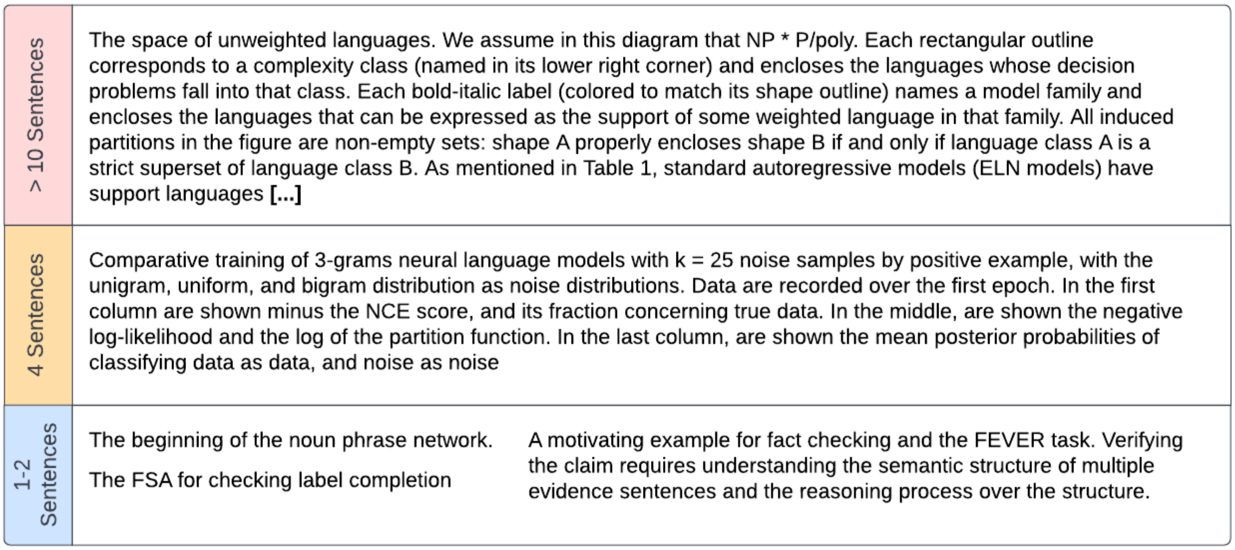


(a) Word count distribution

(b) Sentence count distribution

Figure 3.10: Distribution plots for caption dataset

From previous works [25][7], we know that the sentence length of captions can be highly influential to the learning outcomes of the caption generation model. To take a closer look, we sampled representative captions from various sentence counts (Figure 3.11). We see that for captions with over 10 sentences, the first couple of sentences often offers a short description of the plot (ex: "The space of unweighted languages"). The sentences following gives an extensive description of the "pivot" elements. Those descriptions go into great detail about the methodology and results that the reader can find in the figure. For captioning, the extensive description is not helpful when the model is trying to learn distinctive feature areas in the figure. Instead, most information is contained within the first couple of sentences. Captions with 4-6 sentences have a similar pattern. The results indicated that we needed the preprocessing step to only preserve the first 3 sentences in captions that are longer than 3 sentences.



> 10 Sentences	The space of unweighted languages. We assume in this diagram that NP * P/poly. Each rectangular outline corresponds to a complexity class (named in its lower right corner) and encloses the languages whose decision problems fall into that class. Each bold-italic label (colored to match its shape outline) names a model family and encloses the languages that can be expressed as the support of some weighted language in that family. All induced partitions in the figure are non-empty sets: shape A properly encloses shape B if and only if language class A is a strict superset of language class B. As mentioned in Table 1, standard autoregressive models (ELN models) have support languages **[...]**	
4 Sentences	Comparative training of 3-grams neural language models with k = 25 noise samples by positive example, with the unigram, uniform, and bigram distribution as noise distributions. Data are recorded over the first epoch. In the first column are shown minus the NCE score, and its fraction concerning true data. In the middle, are shown the negative log-likelihood and the log of the partition function. In the last column, are shown the mean posterior probabilities of classifying data as data, and noise as noise	
1-2 Sentences	The beginning of the noun phrase network. The FSA for checking label completion	A motivating example for fact checking and the FEVER task. Verifying the claim requires understanding the semantic structure of multiple evidence sentences and the reasoning process over the structure.

Figure 3.11: Sample caption from different ranges of sentence count

CHAPTER 4

Scientific Figure Captioning

4.1 Experimental Setup

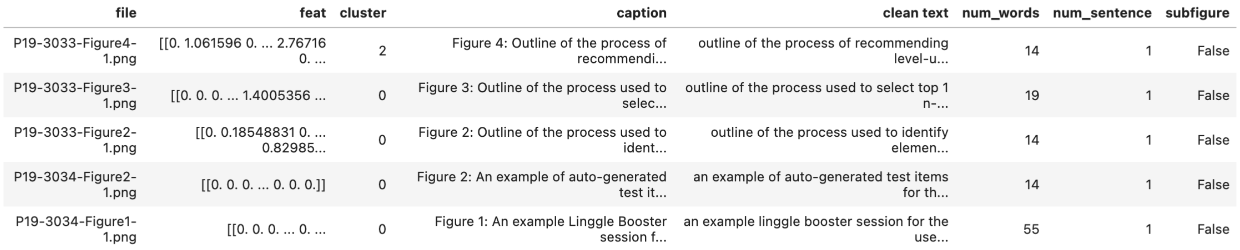
In the previous chapter, we preprocessed and analyzed a large-scale figure-caption dataset. From which we defined clusters representing different types of scientific figures. In this chapter, we will train attention-based caption generation models based on the different figure categories.

4.1.1 Preparing Dataset

The preprocessed captions and figures are stored in their individual datasets, where each element has its figure file name as its unique identifier. We perform an inner join on the file name. Recall that we have 123,721 total figures in our raw dataset. With low-quality data removed from both the figure and caption datasets, our final cleaned dataset contains 119,764 figure-caption pairs. The count of the figure-caption pair for each subgroup defined in the previous chapter is shown in Table 4.1 and an sample of the data is shown in Figure 4.1.

Cluster	Count
Clusters 0 and 4	53,240
Cluster 3	22,459
Clusters 1 and 2	44,065

Table 4.1: Count of data points in each group of the dataset



file	feat	cluster	caption	clean text	num_words	num_sentence	subfigure
P19-3033-Figure4-1.png	[[0. 1.061596 0. ... 2.76716 0. ...	2	Figure 4: Outline of the process of recommendi...	outline of the process of recommending level-u...	14	1	False
P19-3033-Figure3-1.png	[[0. 0. 0. ... 1.4005356 ...	0	Figure 3: Outline of the process used to selec...	outline of the process used to select top 1 n-...	19	1	False
P19-3033-Figure2-1.png	[[0. 0.18548831 0. ... 0.82985...	0	Figure 2: Outline of the process used to ident...	outline of the process used to identify elemen...	14	1	False
P19-3034-Figure2-1.png	[[0. 0. 0. ... 0. 0. 0.]]	0	Figure 2: An example of auto-generated test it...	an example of auto-generated test items for th...	14	1	False
P19-3034-Figure1-1.png	[[0. 0. 0. ... 0. ...	0	Figure 1: An example Linggle Booster session f...	an example linggle booster session for the use...	55	1	False

Figure 4.1: Sample of inner joined figure and caption data

The final processing step for this dataset is to tokenize the cleaned captions. Tokenization is a vectorizing operation in NLP that breaks text strings into smaller word units called tokens. Those tokens will then be mapped to a vocabulary with an assigned index. We employ the Tokenizer method in the TensorFlow text module to perform the tokenization step.

4.1.2 Model Training

In this experiment, we aim to construct three models. Each of the models is trained on one of the defined clusters. The setup for the models is similar. First, we split the data into training and test sets using the 80/20 split. For cluster 3, that means we will have 17,967 data points used for training and 4,492 data points for validation.

We construct the different components of the attention-based caption generation model based on [54]. The first component is a CNN encoder with 256 embedded dimensions. Then we have an RNN decoder employing a Bahdanau attention mechanism [2]. We used the Adam optimizer [28] as the model optimizer and sparse categorical cross entropy as the loss function. We trained the model on a GeForce RTX 2080 Ti GPU. Due to limited time, we trained each model for 20 epochs, where each epoch takes an average of 75 seconds. The loss plot for the model trained on cluster 0 and 4 is shown in Figure 4.2.

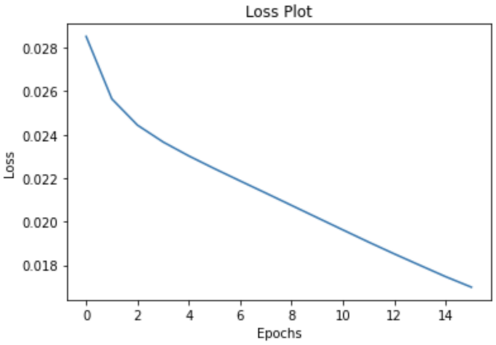


Figure 4.2: Loss plot for caption generation model trained on clusters 0 and 4

4.1.3 Performance Evaluation

After training the model, we evaluate the model in two main ways. First, we compute and record the BLEU, METEOR, ROUGE-L, and Embedding Average Cosine Similarity (EACS) scores using the functions provided by [41]. Due to limited time, we sample 100 data points from the validation set to compute the evaluation metrics, where each iteration takes 37.45 seconds. The second method is plotting the attention grid for the image on every sequence generated by the recurrent decoder. This method is critical when examining the focus of the neural network while captioning the model.

4.2 Results

In this section, we present the evaluation results from our experiment. For each of the three caption generation models, we sampled 2 figures from the results. For those figures, we then show the actual caption, the predicted caption, and the attention grid for each step of sequence generation. Lastly, we provide the evaluation scores on the validation set as described in Section 4.1.3.

4.2.1 Text-based Figures

The first model was trained on clusters 0 and 4, which we have identified in the data analysis section as figures with text-heavy features.

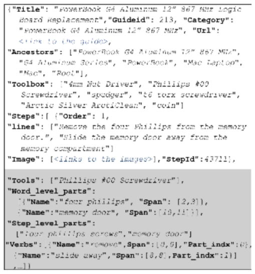
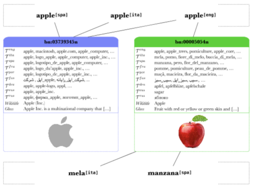
Figure	Original Caption	Predicted Caption
 {"Title": "PowerBook G4 Aluminum 12" 867 MHz Logic Board Replacement","Guideid": 213, "Category": "PowerBook G4 Aluminum 12" 867 MHz", "Url": <link to the guide>, "Ancestors": ["PowerBook G4 Aluminum 12" 867 MHz", "G4 Aluminum Series", "PowerBook", "Mac Laptop", "Mac", "Root"], "Toolbox": ["4mm Nut Driver", "Phillips #00 Screwdriver", "spudger", "t6 torx screwdriver", "Arctic Silver ArctiClean", "coin"] "Steps":[{"Order": 1, "lines": ["Remove the four Phillips from the memory door.", "Slide the memory door away from the memory compartment"] "Image": [<links to the images>],"StepId":43711}, "Tools": ["Phillips #00 Screwdriver"], "Word_level_parts": [{"Name":"four phillips", "Span": [2,3]}, {"Name":"memory door", "Span":[10,11]}], "Step_level_parts": ["four phillips screws","memory door"] "Verbs": [{"Name":"remove",Span":[0,0],"Part_indx":0}, {"Name":"slide away","Span":[0,0],Part_indx":1}] ,...]}	In an example of annotated data from [UNK] the gray area shows the annotation values. only step 1 out of [UNK] in this guide is shown here.	sample dialogue from the annotation scheme
 apple[spa] apple[ita] apple[eng] bn:03739345n bn:00005054n mela[ita] manzana[spa] bn:03739345n T^{eng} apple, macintosh, apple.com, apple_computer, … T^{ita} apple, logo_apple, apple_computer, apple_inc., … T^{spa} apple, logotipo_de_apple, apple_computer, … T^{fra} apple, logo_du'apple, apple_inc., … T^{por} apple, logotipo_de_apple, apple_inc., … T^{fas} شرکت , اپل_apple, شرکت_اپل_رایانه, … T^{deu} apple, apple-logo, appl, … T^{eus} apple, apple_inc. T^{rus} apple, фирма_apple, логотип_apple, … Wikititle Apple (Inc.) Gloss Apple Inc. is a multinational company that […] bn:00005054n T^{eng} apple, apple_trees, pomiculture, apple_core, … T^{ita} mela, pomo, fiore_di_melo, buccia_di_mela, … T^{spa} manzana, pero, flor_del_manzano, … T^{fra} pomme, pomiculture, peau_de_pomme, … T^{por} maçã, macieira, flor_da_macieira, … T^{fas} سیب, سیب_سبز, اپل_سیب, … T^{deu} apfel, apfelblüte, apfelschale T^{eus} sagar T^{rus} яблоко Wikititle Apple Gloss Fruit with red or yellow or green skin and […]	retrieval of two senses for five seed terms in three different languages.	a flow chart of the example sentence in [UNK]

Table 4.2: Sample figures with original and predicted captions from clusters 0 and 4

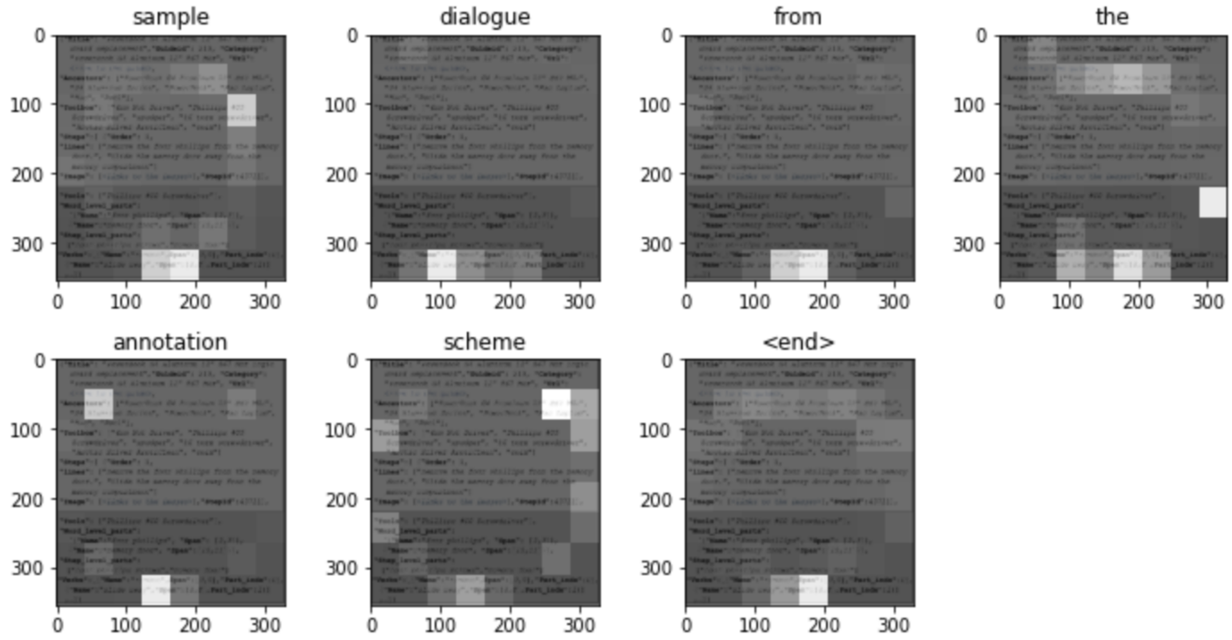


Figure 4.3: Attention grid for sample figure 1 in clusters 0 and 4

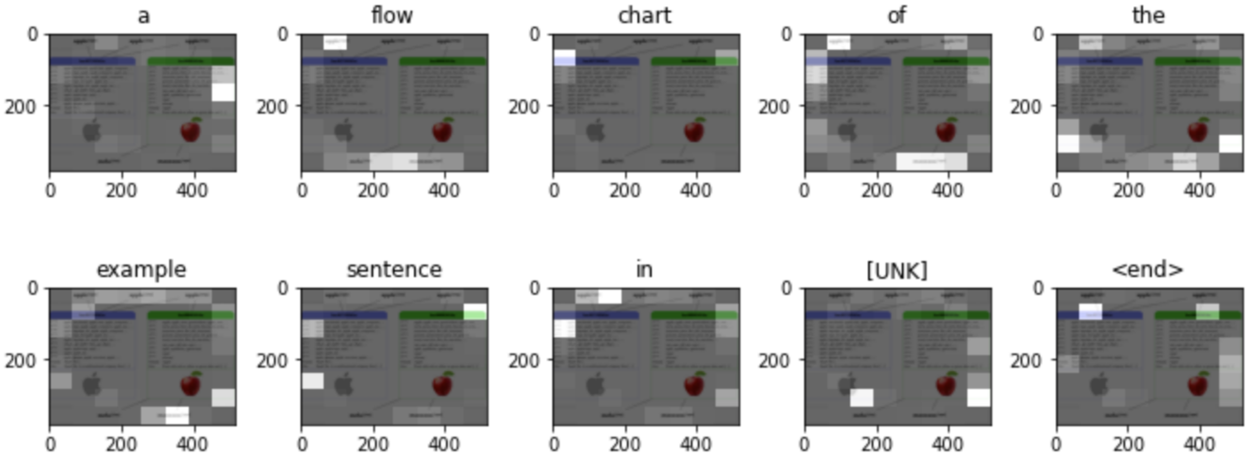


Figure 4.4: Attention grid for sample figure 2 in clusters 0 and 4

4.2.2 Block-based Graphs

The second model is trained on clusters 1 and 2, which we identified as figures that have block-like or cluttered features such as bar graphs and histograms.

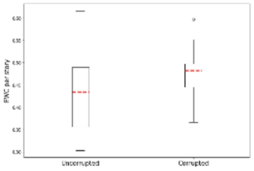
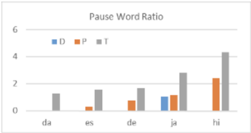
Figure	Original Caption	Predicted Caption
	indices of coherence derived from human participant evaluation of [UNK] [UNK] lower is better.	distribution of mturk workers [UNK] [UNK]
	comparison of [UNK] across different languages	comparison of performance of [UNK] and multilingual model development task, data, training sets

Table 4.3: Sample figures with original and predicted captions from clusters 1 and 2

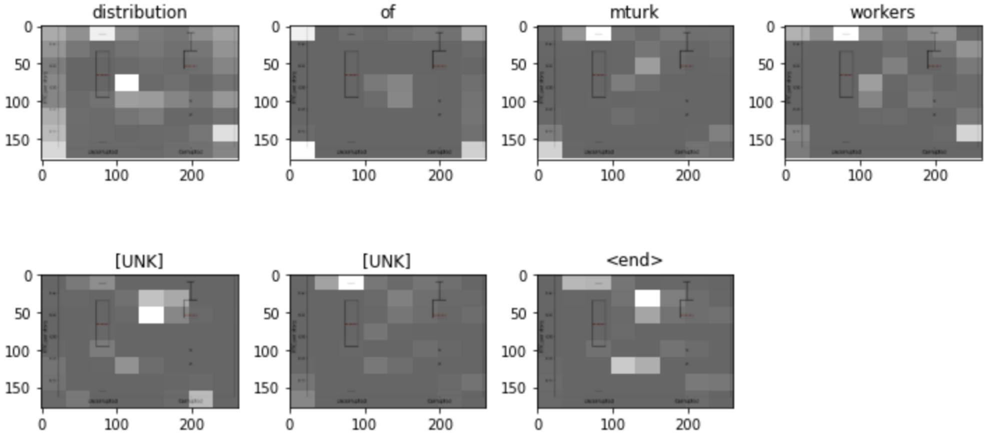


Figure 4.5: Attention grid for sample figure 1 in clusters 1 and 2

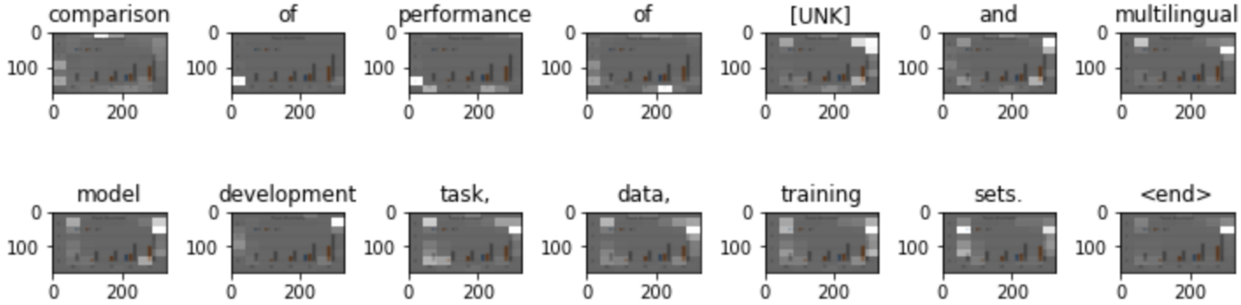


Figure 4.6: Attention grid for sample figure 2 in clusters 1 and 2

4.2.3 Line-based and Sparse Graphs

The third model is trained on cluster 3, which we identified as figures that have line-based or sparse features such as line graphs and scatter plots.

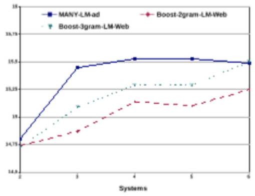
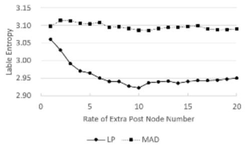
Figure	Original Caption	Predicted Caption
	comparison of n-gram boost approaches.	unsupervised [UNK] f1 score on x-axis.
	label entropy of [UNK] with different rate of extra post nodes	[UNK] i for different training with finetuning performance

Table 4.4: Sample figures with original and predicted captions from cluster 3

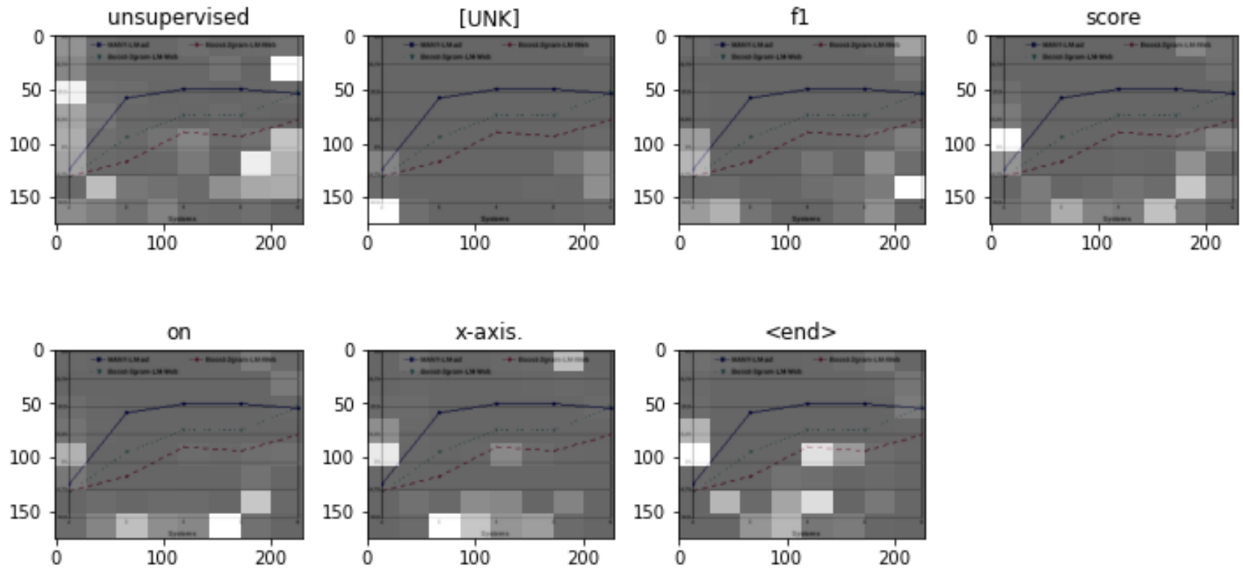


Figure 4.7: Attention grid for sample figure 1 in cluster 3

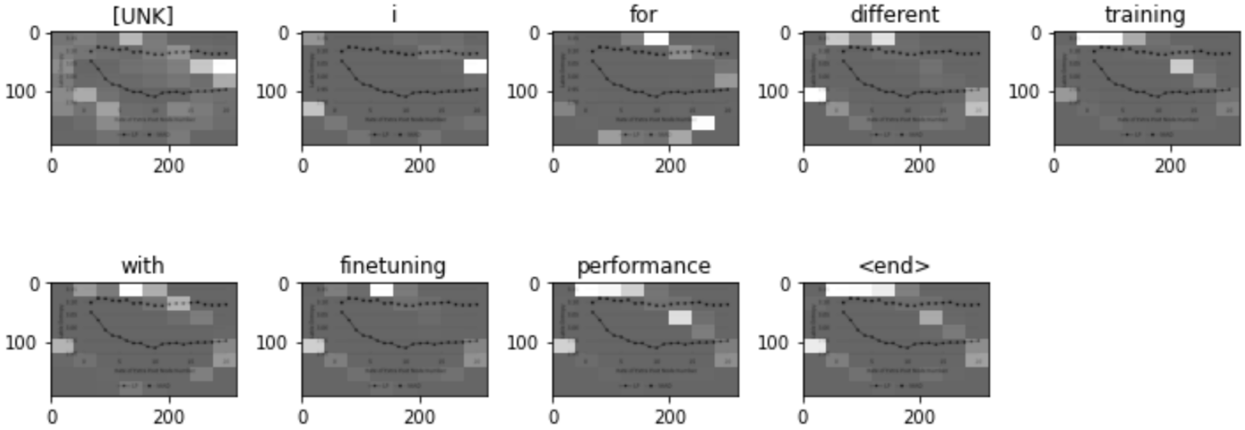


Figure 4.8: Attention grid for sample figure 2 in cluster 3

4.2.4 Evaluation

Table 4.5 presents the evaluation scores for the three models labeled by the clusters they were trained on. Noting again that the scores are calculated based on 100 data points from the validation set. A detailed description of the scores can be found in Section 2.2.4.

Model	BLEU-4	METEOR	ROUGE-L	EACS
Clusters 0 and 4	0.285	0.179	0.204	0.801
Clusters 1 and 2	0.003	0.163	0.197	0.815
Cluster 3	0.001	0.148	0.185	0.817

Table 4.5: Evaluation Scores for experiments on caption generation models

4.3 Discussion

In our experiment, we trained our attention-based caption generation model on three subsets of our dataset. We empirically determined the majority of figure types in those subset clusters as text-based, block-based, and line or sparse-based. We presented the results from each of the subsets with sample figures, their predicted caption, attention grid, and evaluation metrics. Generally, the predicted captions are different in terms of vocabulary compared to the original caption but highlight similar "pivot" elements. For example, the original caption from figure 1 of Table 4.4 described it as a high-level comparison of experiment outcomes. The caption on the other hand described the graph's features such as the legend and x-axis. In another case, such as Figure 2 from Table 4.3, the predicted and original caption both captured the comparative nature of the figure.

One possible reason could be because block-based figures such as bar graphs are more often used for comparison, and the model picked up on that trend when training on the captions. The results showed that captioning models can capture semantic information in a similar fashion as humans, however, the dataset has to be treated such that the generated captions are more normalized in terms of focus areas.

We expected that figures from each subset will have different "pivot" elements and attention areas. The behavior can be observed with the attention grids. The text-based figures are difficult to caption due to the lack of visual pivot elements. For other figures, the models use the text in axis labels and legends to gain a clue about the captioning which are clearly not available for text-heavy figure such as Figure 4.3. The results for text-based figures showed that the models in turn learn to recognize general objects that describe the figure. In Figure 4.3, the model recognized that it is an annotation scheme. However, it falsely captioned the JSON-like data as dialogue, possibly due to the quotes. The same thing happens with Figure 4.4. The model recognized the boxes and lines as a flow chart, which is an abundant figure type in clusters 0 and 4, wherein reality it is a labeled comparison plot. From the second and third instance, we can see the model showing heavy attention on top to bottom text ("flow") and box containers ("chart"). The results showed that captioning is difficult for text-based figures, since different categories of those figures share the same visual features.

Block-based figures, unlike text-based figures, have more diverse visual features, and text serves as pivot element. In Figure 4.5, the model was able to first recognize that the overall plot is illustrating some type of distribution. Then, the third and fourth attention instance was able to utilize the axis to determine the subject of interest, which was captioned as Amazon Mechanical Turk (mturk) workers. Although the original caption uses an entirely different vocabulary, where they called the subjects human participants, the important message is conveyed in both cases. The theme is similar for Figure 4.6. The model was able to capture the fact that the bars serve as a comparison and that it is comparing different languages.

Line-based figures has a similar proportion of visual feature to text as block-based figures. However, the visual features in those figures are much harder to distinguish, and the purpose of those figures are usually more diverse. The results showed that the model was able to describe the figure in both samples, but the scientific narrative wasn't preserved in the same way as the original caption. Figure 4.7 is a unique case where the model specifically described the label of the x-axis. This is possibly due to training captions that also describe the axis of plots, different from the approach taken from the original caption of this particular figure. This suggests that apart from separating the figures, the caption length can have a influence on model performance [25]. Further experiments can be performed to analyze if normalizing caption structure and intention can enhance the model.

We were able to analyze in detail the model behavior by looking at attention grids, but evaluation metrics allow us to have a better look at the overall performance based on the selected metrics. Our results showed that the model for clusters 0 and 4 has the highest BLEU-4, METEOR, and ROUGE-L scores and the lowest EACS score. BLEU-4 and METEOR were originally developed for machine translation. Since clusters 0 and 4 are text-based, it can be expected that the vocabulary similarity between the generated caption and the original caption is similar, leading to a higher score in those metrics. On the other hand, the other models that are better at describing the same object with different vocabulary have a higher EACS score, which measures the embedded meaning of the words. For comparison, the results in [25] which are also trained on extract graphs have an average BLEU-4 score of 0.021.

In summary, our results showed that the attention-based figure captioning model can generate grammatically correct captions that describe the figures. Between the three categories of figures we defined, the block-based figures with distinct visual features and a small function pool provided the highest quality caption that convey a scientific narrative. Those conclusions align with the hypothesis provided in previous research [25][7]. Some standard evaluation scores showed that the current model has unsatisfactory performance in most cases. Further sub-setting the figures and manually identifying caption purposes are potential ways to significantly improve model performance that can be explored in future research. In Section 5.1, detailed suggestions on future experimentation for this topic will be described.

CHAPTER 5

Conclusion

In this work, we investigated caption generation for large-scale real-world scientific figures. We provided extraction and analysis of the figures and metadata from the Association of Computational Linguistics. This large dataset is then cleaned and clustered. The clustering analysis produced 5 clusters and 3 empirically defined figure categories. Those categories are then used to train three individual captioning models for which we examined the performance. Our experimental results showed that the model achieve best performance when captioning block-based figures due to their diverse visual features. The analysis was carried out by plotting the attention grid generated by our attention-based models. Our results can be helpful for future research by highlighting the opportunities and challenges of captioning extracted figures and providing the analytical tools that can be used to evaluate those data and models.

With captioning being an integral portion of any scientific literature, producing quality captions automatically can be highly influential. Researchers can leverage automatic captioning to assess the quality of their captions and seek recommendation. Users with disability can use automatic captioning to receive scientific message in wild figures. Although there are steep challenges for this task, as presented in this work, novel methods and high quality data can be developed in the near future to empower communication of scientific knowledge.

5.1 Future Work

The complexity for the data used in this work comes from it being automatically extracted. The large scale and diverse data types also adds to that complexity. Since the data is in many cases integral to the performance of machine learning models, extra time and resources can be used to increase the quality of this dataset. Some cleaning procedure used in this work includes removing sub-figures by recognizing patterns in captions, removing outliers and redundant components in captions, and cropping extra information from long captions. More cleaning can be done by employing both automated and manual procedures. For example, one can automatically filter sub-figures with FigureSeparator [49]. For the captions, one might be able to build a rule-based system that only preserve captions that is defined by a standard format. This way, there won't be captions that only provide a high-level description while others go into detailed context. After low quality data is removed with those procedures, the dataset might be scaled down enough for manual cleaning and inspection which is highly desired.

In our analysis, we adopted a clustering approach to segment our data due to the limited knowledge on the types of figures in the ACL corpus. To expand on this work and gain more sophisticated insight, one can use a different clustering technique such as DBSCAN [15] or choose different number of clusters. An effective strategy used in previous research is to segment the figures using a figure classifier [25]. From our clustering analysis, we found that a significant amount of data has overlapping. By extracting the data closest to the centroid of each cluster can improve the separation. For the captions, the sentence tokenizer in some cases miscalculated the sentence count due to the common use of periods (.) in figure captions. A more sophisticated tokenizer can be built to resolve that issue. Lastly, a CNN model other than VGG16 can be used to extract figure features.

We expect that training the model on higher quality data will improve its performance. However, there is still some tasks that can be extended on this work. Due to limited resources, the model wasn't trained and validated fully. We only trained the model for 20 epochs, which can be increased. We ran our validation on 100 data points while we have more than 1000 examples for validation in each cluster. One can also train the model on more specific figure types with consistent caption in a way similar to [25]. Additional features such as text and vision context could be added to adapt to the inconsistency in real-world figure data. Lastly, one can always experiment with other model architectures.

Bibliography

[1] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic propositional image caption evaluation. In *ECCV*, 2016.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2015.

[3] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *IEEvaluation@ACL*, 2005.

[4] Yoshua Bengio, Patrice Y. Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5 2:157–66, 1994.

[5] Steven Bird. Nltk: The natural language toolkit. *ArXiv*, cs.CL/0205028, 2004.

[6] Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the role of bleu in machine translation research. In *EACL*, 2006.

[7] Chen Chen, Ruiyi Zhang, Eunyee Koh, Sungchul Kim, Scott D. Cohen, and Ryan A. Rossi. Figure captioning with relation maps for reasoning. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1526–1534, 2020.

[8] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6298–6306, 2017.

[9] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*, 2014.

[10] Colin B. Clement, Matthew Bierbaum, Kevin P. O'Keeffe, and Alexander A. Alemi. On the use of arxiv as a dataset. *ArXiv*, abs/1905.00075, 2019.

[11] Wikimedia Commons. File:artificial neural network.svg — wikimedia commons, the free media repository, 2020. [Online; accessed 28-February-2022].

[12] Wikimedia Commons. File:convolutionandpooling.svg — wikimedia commons, the free media repository, 2020. [Online; accessed 28-February-2022].

[13] Wikimedia Commons. File:perceptron-unit.svg — wikimedia commons, the free media repository, 2020. [Online; accessed 28-February-2022].

[14] Chris DeBrusk and Oliver Wyman. The risk of machine learning bias (and how to prevent it). 2020.

[15] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, 1996.

[16] Hao Fang, Saurabh Gupta, Forrest N. Iandola, Rupesh Kumar Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C. Platt, C. Lawrence Zitnick, and Geoffrey Zweig. From captions to visual concepts and back. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1473–1482, 2015.

[17] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, J. Hockenmaier, and David Alexander Forsyth. Every picture tells a story: Generating sentences from images. In *ECCV*, 2010.

[18] Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 1*, 2:559–572.

[19] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 2004.

[20] Aurélien Géron. Hands-on machine learning with scikit-learn and tensorflow: Concepts, tools, and techniques to build intelligent systems. 2017.

[21] Tayfun Gokmen, Malte J. Rasch, and Wilfried E. Haensch. Training lstm networks with resistive cross-point devices. *Frontiers in Neuroscience*, 12, 2018.

[22] Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. Deep learning. *Nature*, 521:436–444, 2015.

[23] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

[24] Md. Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CSUR)*, 51:1 – 36, 2019.

[25] Ting-Yao Hsu, C. Lee Giles, and Ting-Hao Kenneth Huang. Scicap: Generating captions for scientific figures. In *EMNLP*, 2021.

[26] Xu Jia, Efstratios Gavves, Basura Fernando, and Tinne Tuytelaars. Guiding the long-short term memory model for image caption generation. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2407–2415, 2015.

[27] Junqi Jin, Kun Fu, Runpeng Cui, Fei Sha, and Changshui Zhang. Aligning where to see and what to tell: image caption with region-based attention and scene factorization. *ArXiv*, abs/1506.06272, 2015.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

[29] Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C. Berg, and Tamara L. Berg. Babytalk: Understanding and generating simple image descriptions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35:2891–2903, 2013.

[30] Daniel T. Larose and Chantal D. Larose. An introduction to data mining. 2014.

[31] Siming Li, Girish Kulkarni, Tamara L. Berg, Alexander C. Berg, and Yejin Choi. Composing simple image descriptions using web-scale n-grams. In *CoNLL*, 2011.

[32] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *ACL 2004*, 2004.

[33] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3242–3250, 2017.

[34] J. MacQueen. Some methods for classification and analysis of multivariate observations. 1967.

[35] Adam H. Marblestone, Greg Wayne, and Konrad Paul Körding. Toward an integration of deep learning and neuroscience. *Frontiers in Computational Neuroscience*, 10, 2016.

[36] Tom M. Mitchell. *Machine learning, International Edition*. McGraw-Hill Series in Computer Science. McGraw-Hill, 1997.

[37] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2002.

[38] Marco Pedersoli, Thomas Lucas, Cordelia Schmid, and Jakob Verbeek. Areas of attention for image captioning. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1251–1259, 2017.

[39] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015.

[40] Rohan Sethi and Ila Kaushik. Hand written digit recognition using machine learning. *2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT)*, pages 49–54, 2020.

[41] Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *ArXiv*, abs/1706.09799, 2017.

[42] Noah Siegel, Nicholas Lourie, Russell Power, and Waleed Ammar. Extracting scientific figures with distantly supervised neural networks. *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries*, 2018.

[43] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015.

[44] Chiranjibi Sitaula and Mohammad Belayet Hossain. Attention-based vgg-16 model for covid-19 chest x-ray image classification. *Applied Intelligence (Dordrecht, Netherlands)*, 51:2850 – 2863, 2021.

[45] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. *ArXiv*, abs/1906.02243, 2019.

[46] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014.

[47] Ken Thompson. Programming techniques: Regular expression search algorithm. *Commun. ACM*, 11:419–422, 1968.

[48] Kenneth Tran, Xiaodong He, Lei Zhang, and Jian Sun. Rich image captioning in the wild. *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 434–441, 2016.

[49] Satoshi Tsutsui and David J. Crandall. A data driven approach for compound figure separation using convolutional neural networks. *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 01:533–540, 2017.

[50] Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[51] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4566–4575, 2015.

[52] Oriol Vinyals, Alexander Toshev, Samy Bengio, and D. Erhan. Show and tell: A neural image caption generator. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164, 2015.

[53] Chaoyang Wang, Ziwei Zhou, and Liang Xu. An integrative review of image captioning research. *Journal of Physics: Conference Series*, 1748, 2021.

[54] Ke Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015.

[55] Zhilin Yang, Ye Yuan, Yuexin Wu, Ruslan Salakhutdinov, and William W. Cohen. Encode, review, and decode: Reviewer module for caption generation. *ArXiv*, abs/1605.07912, 2016.

[56] Alice Zheng, Nicole Shelby, and Ellie Volckhausen. Evaluating machine learning models. *Machine Learning in the AWS Cloud*, 2019.

[57] Y. T. Zhou and Rama Chellappa. Computation of optical flow using a neural network. *IEEE 1988 International Conference on Neural Networks*, pages 71–78 vol.2, 1988.

APPENDIX A

Python Code

The Python codes that perform the data cleaning, analysis, and captioning model construction can be found in this link: https://github.com/billchen0/scientific-figure-captioning.git. The experiments and data can be recreated from the code provided.

Bill Chen

(billchen0011@gmail.com | linkedin.com/in/billchen0011

Education

B.S. Engineering Science and Mechanics
Pennsylvania State University - Schreyer Honors College - Graduating May 2022
Honors/Awards: - Dean's list, Academic Excellence Scholarship, Student Engagement Grant

Experiences

Nittany Ai Alliance - Data Scientist January 2022 - May 2022

- Led development of an application that reduces negative effects of social media on teenager health.

Johnson & Johnson | Ethicon Inc. - Data Science Co-op January 2021 - May 2021

- Built a data pipeline to retrieve and format data from a REST API and an Amazon Redshift database.
- Cleaned and formatted advanced energy device data and surgical outcome data using Python.
- Created visualizations to derive insights for surgeons and device researchers using Tableau.
- Communicated with surgeons and engineers to develop a dashboard product through an agile process.

PsySpace - Co-founder - Team Lead October 2020- Present

- Applying NLP techniques to analyze mental health data and build predictive language models.
- Trained a multi-label multi-class text classification model using Tensorflow and deployed it to Google Cloud Platform.
- Performed web scraping and survey creation for training data collection with Python.
- Leading a team of 6 developers, 12 psychology specialists, and 2 UX designers to create a mobile application.
- Leading important business functionality teams such as Marketing, PR, and HR.
- Finalist at the 2021 Nittany AI Challenge | Received $10,000 in funding

Intelligent Systems Research Laboratory - Undergraduate Researcher January 2021 - Present

- Building the next generation CiteSeerX with Vue.js, FastAPI, and ElasticSearch.
- Researching and developing state-of-art entity disambiguation methods to enhance intelligence features.
- Thesis: Scientific Figure Captioning with Visual Attention Models

Cocoa Packs | Nittany Data Lab - Software Engineering Intern May 2020 - August 2020

- Developed a web application for the non-profit organization to manage their volunteer, recipient, and donor profiles using the Django framework.
- Created data reports helping managers gain insight into various distribution activities.

Projects

Brain Tumor Localization - Classified and localized brain tumors in MRI images with state-of-art computer vision techniques.
Personalized Medicine - Analyzed NCBI lung tumor clinical and genetic data to examine targets for personalized medicine.

Skills

- **Programming:** Python, R, JavaScript, Dart, C++, MATLAB, SQL, HTML, CSS, LaTeX, Markdown
- **Data Science:** Tableau, Numpy, Pandas, Matplotlib, Seaborn, Tensorflow, Keras, PyTorch
- **Development:** Flutter, Flask, Vue.js, Django, ElasticSearch, React Native, Google Cloud, AWS, Firebase
- **Tools:** Git, Github, Visual Studio Code, Jupyter
- **Others:** Microsoft Office, Google Suite, English & Chinese (native proficiency)