SMU Data Science Review

Volume 3 | Number 1 Article 2

2020

Improving Syntactic Relationships Between Language and Objects

Benjamin Wilke
Southern Methodist University, bwilke@smu.edu

Tej Tenmattam
Southern Methodist Univeristy, ttenmattam@smu.edu

Anand Rajan
Southern Methodist University, anandr@smu.edu

Andrew Pollock
Getty Images, andrew.pollock@gettyimages.com

Joel Lindsey
Southern Methodist University, jdlindsey@smu.edu

Improving Syntactic Relationships Between Language and Objects

Benjamin Wilke[1], Tej Tenmattam[1], Anand Rajan[1], Andrew Pollock[2], and Joel Lindsey[1]

[1] Master of Science in Data Science, Southern Methodist University, Dallas, TX 75275 USA {jdlindsey,anandr,ttenmattam,bwilke}@smu.edu
[2] Getty Images, New York, NY 10007 USA {Andrew.Pollock}@gettyimages.com https://www.gettyimages.com/

Abstract. This paper presents the integration of natural language processing and computer vision to improve the syntax of the language generated when describing objects in images. The goal was to not only understand the objects in an image, but the interactions and activities occurring between the objects. We implemented a multi-modal neural network combining convolutional and recurrent neural network architectures to create a model that can maximize the likelihood of word combinations given a training image. The outcome was an image captioning model that leveraged transfer learning techniques for architecture components. Our novelty was to quantify the effectiveness of transfer learning schemes for encoders and decoders to qualify which were the best for improving syntactic relationships. Our work found the combination of ResNet feature extraction and fine-tuned BERT word embeddings to be the best performing architecture across two datasets - a valuable discovery for those continuing this work considering the cost of compute for these complex models.

1 Introduction

Simple object identification has become widely available through common pretrained image classification deep learning architectures and is even available as a ready-made cloud service on Amazon Web Services and Google Cloud Platform. These architectures are designed to report on the presence of an object or return a bounding box that designates the object position and size. Our goal is to move to higher order relationships of the objects in an image as understood through natural language descriptions. Our work will focus on descriptive adjectives about an object (a noun) itself and prepositions that describe proximity between objects. For example - we are interested in determining adjectives describing the size or color of a single object: "large cat", "blue ball", and "red sign" and prepositions or prepositional phrases describing the relationships between two or more objects: "boy inside car", "grass outside window", and "statues on top of a building". We are also interested in preserving the correct context and adjective relationships when applying our labels. For example, a label associated

with "soccer player wearing a blue jersey and red socks" should not be applied to an image with a person wearing a red jersey and blue socks. Our project sponsor Getty Images identified the disambiguation of image search terms as one of the most challenging questions for image brokers where market differentiation for their web presence is driven solely by image search.

Our approach to solve this problem was to create an image captioning model that successfully blends natural language processing and computer vision in an encoder/decoder deep neural network. Our model jointly learned specific relationships of image descriptions to the contents of an image. Once trained the model was capable of generating the next most probable word given the contents of the image and previous words until a full sentence was formed. This problem is very challenging to the machine learning community as it combines two domains that are in themselves very challenging. These domains are very computationally expensive, which makes incremental updates to architectures and model hyperparameters very hard to quantify (as updated models are very expensive to train). While the application of transfer learning has lowered the barrier to entry for performing these tasks, it's not known which combinations of architectures yield the best results. Thus our novelty is to understand which combinations produce the most accurate results for our problem statement above. Ultimately, we found that combining ResNet image feature extraction and pretrained BERT word embeddings produced high performing results when measured on our datasets.

In Section 2 we will outline the related work that inspired our further research and we frame the complexity of image captioning tasks. Section 3 provides a primer on natural language processing from a practical sense as well as the applications of modern deep learning techniques to this domain. We will provide a brief overview of word embeddings and the innovations brought about by GloVe and BERT. Section 4 outlines the various Convolutional Neural Networks (CNNs) that we will apply to our image encoder. We give a light primer on how CNNs provide value to our research and the distinctions between the various architectures that we employed, including: VGG-16, VGG-19, ResNet, and Inception. Our research utilized transfer learning for both our image encoder and word embeddings, which we outline in Section 5. We will discuss how transfer learning is applied to our models and the nuance between frozen versus fine-tuned applications. Section 6 discusses Recurrent Neural Networks (RNN) and Long Short Term Models (LSTM) and their benefit to natural language generation required for an image captioning task. Section 7 discusses the data sets employed in our research, including: Flickr8k and Microsoft COCO. In Section 8 we bring our multi-modal image captioning model together and discuss our process for preparing our data and the complexities we encountered for compute and memory capacity. Section 9 discusses measuring the effectiveness of each of our models through the application of a scoring technique known as Bilingual Evaluation Understudy Score (BLEU). This section also highlights our results and future work. Section 10 outlines some of the ethical considerations we encountered when considering our research before concluding in Section 11.

2 Related Work

Our work builds from two complementary papers on image caption generation [1] and [2]. The task of image caption generation has been explored using solutions that attempt to merge the sub-tasks of object recognition and description, but it wasn't until recently that joint models optimized for these combined tasks emerged. The earliest inspiration was found in the advancement of machine translation to take advantage of RNNs configured in a encoder/decoder architecture. This discovery was an advancement from simpler approaches, which included naively translating words individually, without regard for surrounding context. The proposal of [1] suggested that the encoder RNN could simply be swapped with a CNN. The CNN would now generate an encoded vector representation of image contents to pass into the hidden layer of an RNN decoder to produce a "translation" - in this case an English description from an image. The challenge became efficiently training the encoder/decoder network as well as recognizing more complex interactions between objects in an image.

The authors of paper [2] innovated on this concept through the application of attention. CNNs are designed to benefit from reducing noise and clutter in an image to the most salient objects for simple object identification. However, the issue with applying these methods and compressing an entire image down to a single encoding is often lost in the interactions between objects. It's clear that a thoughtful algorithm should preserve this information by examining the low-level representations of an input image, and to steer the algorithm towards information that is relevant in the image. This concept was initially proposed in paper [3], which shows that compressing all the necessary information into a fixed length vector not only includes parts of the inputs that may not be important, but poses problems for inputs (sentences in this case) that are longer than any of the sentences in the training corpus. "Bahdanau attention" - named for his contributions - most importantly proposes an encoder/decoder model that learns to align and translate jointly. That is, for each word generated in the decoder translation a soft-search is performed for areas of the source sentence where the most relevant related information can be found. The authors [2] demonstrated how Bahdanau attention [3] can be applied to images.

The authors of [1] of [2] both utilized VGG-16 pre-trained weights for their image encoding task and discussed the opportunities for initializing their word embeddings (but were unsuccessful, so left them uninitialized). It's clear how our work to evaluate several pre-trained CNN architectures and word embeddings will be important for future image captioning tasks.

3 Natural Language Processing

Natural Language Processing (NLP) refers to the development and use of machine learning methods for processing, understanding, and generating natural language. NLP models do not understand language in the human sense, but rather they are trained to understand an imposed statistical structure of language [4]. Once the statistical structure of language has been learned it can be

used to understand new language inputs or produce new language outputs from arbitrarily trained inputs (other languages, or in our case image representations).

3.1 Word Embeddings, Word2Vec, and GloVe

Word embeddings provide a means to produce dense feature representation, while also preserving word relationships. Most critically, these embeddings are not manually encoded, but rather they are learned as part of the model training [5]. Google paved the way for word embedding models with their release of Word2Vec in 2013 [6]. The Stanford authors [7] and creators of Global Vectors for Word Representation (GloVe) improved on the Word2Vec approach to produce word embeddings by utilizing a global word-word co-occurrence matrix. GloVe is trained on Wikipedia 2014 and Gigaword 5 and the corpus included over 6 billion tokens and has a 400,000 uncased vocabulary. Our models utilized the 200-dimension embedding vectors provided by [7] and as these embeddings were used in their pre-trained state the implementation cost was quite low.

3.2 BERT

GloVe word embeddings are context-free representations with only a single representation for each word in the vocabulary. State of the art contextual models such as Bidirectional Encoder representations for Transformers (BERT) were introduced into NLP to solve this problem by generating rich representations of words based on other words in their sentence context.

Google's transformer model BERT [8] gained popularity in the NLP community in 2018 and is now known to perform NLP tasks with cutting edge results. BERT utilizes transformers, which consist of multiple layers where each layer contains multiple attention heads. Since BERT transformers read a sequence entirely at once, the sequence is learned bidirectionally and important contextual words further away from key words maintain importance.

BERT added a layer to our core model and this layer needed fine-tuning for our image captioning task (fine-tuned to each dataset). The BERT architecture (as shown in Figure 1) is complex and highlights a theme in modern NLP that many tasks can be improved at the expense of model complexity. Our fine-tuning task took over 30 hours for each dataset on a leading Google Cloud Compute GPU-enabled instance. The newest innovations on BERT, such as ALBERT [9], focus on reducing the complexity of the transformer architecture while maintaining its distinction as the most powerful tool in modern NLP.

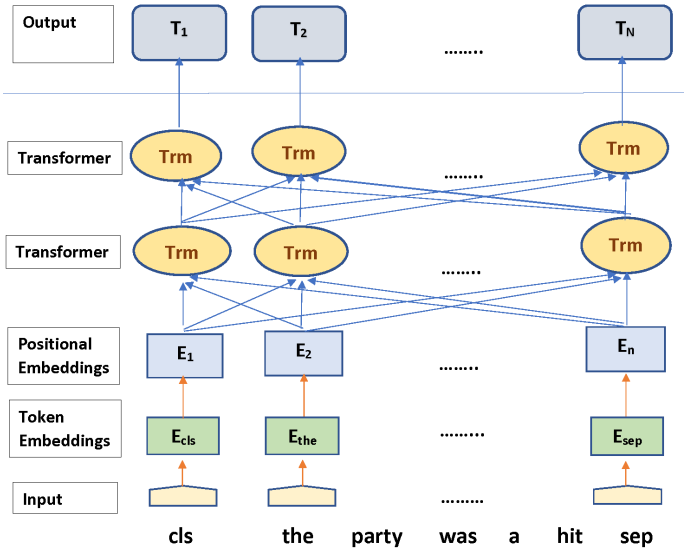

Fig. 1. Bidirectional Encoder Representations for Transformers (BERT) Architecture

4 Convolutional Neural Networks for Image Classification

Deep learning applications for image analysis and classification through the use of CNNs has made many significant advances in recent years as access to cloud computing and graphics processing units (GPU) has become more widespread. Many pre-trained CNN architectures are available that we can apply through transfer learning (described in Section 5). For our CNN models, we selected: VGG-16, VGG-19, ResNet and Inception. All of these models have been pre-trained on the ImageNet dataset, which consists of 15 million labeled images belonging to almost 22,000 categories.

4.1 VGG-16

The VGG models for image classification were developed by the Oxford Visual Geometric Group [10]. VGG-16 became renowned upon winning the 2014 ImageNet Large-Scale Visual Recognition Challenge (ILSVRC). Figure 2 depicts the VGG-16 architecture and processing steps.

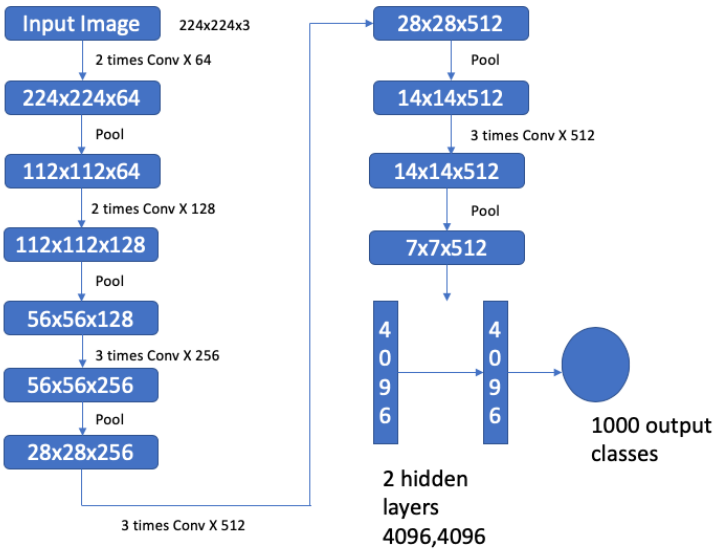

Fig. 2. VGG-16 Architecture

4.2 VGG-19

VGG-19 was also proposed in [10]. Unfortunately, both VGG architectures suffered from two major drawbacks. The first is that they were both very computationally expensive to train. The second issue is that once they were trained the resulting model weights were hefty considering the size of the architectures. This made the deployment of VGG difficult and reduced practical portability in terms of disk space and network bandwidth.

VGG-19 increased the depth of the network when compared to VGG-16, which also improved the accuracy. However, additional accuracy gains are stifled by the vanishing gradient. As the number of layers in a network increases, the calculated gradient provided by back propagation becomes very small in early network layers and the network performance deteriorates.

4.3 ResNet

Residual Network (ResNet) addresses this specific scenario. ResNet requires that the convolution layer takes the previous layers output as input and learns the additional features that need to be learned to perform classification. The term "residual" is the additional feature that the model is expected to learn from one layer to the next layers. The ResNet architecture in Figure 3 suggests that extremely deep networks can be improved through the use of residual modules.

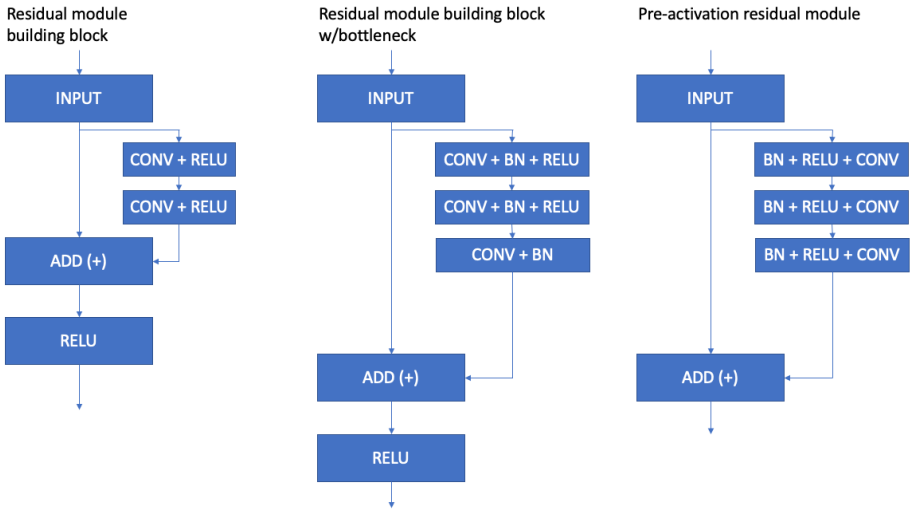

Fig. 3. ResNet Architecture

Resnet was first introduced in [11] and has become an important work in the deep learning literature. The authors [11] were able to further increase the accuracy by using identity maps outlined in [12].

4.4 Inception

GoogleNet (Inception v1) was first introduced in [13] and subsequently improved to Inception v2 in [14].

The innovation brought about by Inception stems from applying not one, but three different size filters to the same input versus progressive application. An Inception module is shown in Figure 4. Inception-based networks are evolving quickly and popular versions include v1, v2, v3, v4, and Inception-ResNet. Our architecture utilized Inception v3.

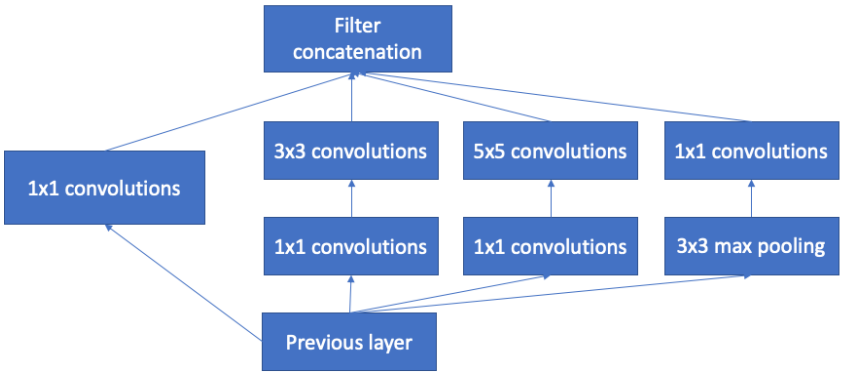

Fig. 4. Inception Architecture

5 Transfer Learning

Transfer learning emerged from the discovery that initial layers in deep neural networks are not specific to any single task, but generalize to all tasks once appropriately trained. It's been shown that this practice will not only save computation, but may also improve generalization across disparate datasets [15]. For example - the authors [15] found that a CNN trained for the detection of natural entities (animals, plants) could perform a new task of detecting man-made entities (buildings, vehicles) with little tuning. Transfer learning can be implemented in two ways depending on project requirements: using the pre-trained architecture in a locked state or fine-tuning. The distinction is whether you allow the pre-trained weights to be adjusted to a new task or you lock them from backpropagation during new task specific training [4].

Our first application of transfer learning was using pre-trained CNNs as fixed feature extractors as outlined in Section 4. Our goal was to compare the architectures so we used the frozen pre-trained weights. Technically, there was no opportunity for fine-tuning these architectures as the extracted image features were used in combination with our recurrent language model outlined in Section 8. Our merged model [16] learned these joint combinations, which occurred after the raw image feature extraction.

Our second application of transfer learning was our application of word embeddings discussed in Section 3. We implemented frozen GloVe embeddings as released by [7], which may leave opportunity for future work to explore fine-tuning the GloVe embeddings. We chose to fine-tune our BERT embeddings on our corpus (all image captions per dataset) as BERT is a contextualized language model and benefits from understanding the specific context in our captions. This

also leaves opportunity for future work to explore how well the existing BERT embeddings [8] may perform in their base state.

6 Recurrent Neural Networks

Modern deep learning practitioners agree that the proposal of a deep neural network emulating the human brain is fairly absurd [4]. This becomes even more apparent considering that the human brain has capacity to remember short and long term dependencies that are related to inputs in real time. The role of RNN architectures is to address these dependencies. This concept diverges from a standard feed-forward neural network architecture where inputs are converted to outputs by a fixed transformation and is a requirement for our language generation models. All RNN architectures have the form of repeating modules (Figure 5) where each sequence step considers the current step inputs and recurrent outputs of all previous steps.

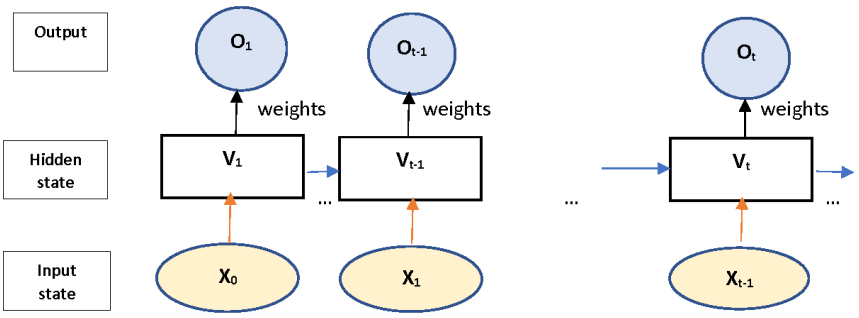

Fig. 5. Simple RNN

Our models utilized a LSTM layer (consisting of LSTM cells shown in Figure 6) which are highly effective for sequence to sequence learning like our image captioning task. LSTMs have a chain like structure with layers interacting like a conveyor belt, with the capability to add and remove information to each cell state through gates which optionally filter or forward information. [17].

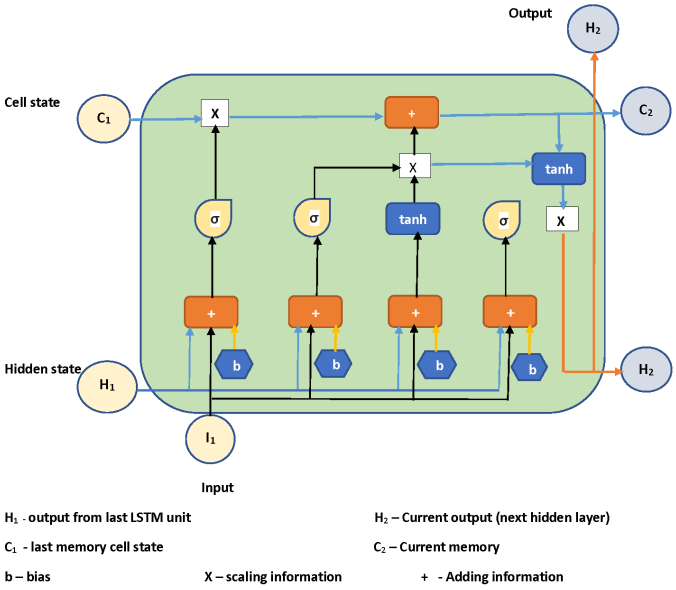

Fig. 6. LSTM Cell Architecture

7 Datasets

Our initial models utilized the Flickr8k dataset, which consists of 8,000 images that are each paired with 5 different captions providing clear descriptions of image entities and interactions. These images were chosen from 6 different Flickr groups and were manually selected to maximize the variety of scenes and situations. The Flickr8k dataset was an excellent choice for initial discovery as it could be used to train models on basic hardware (like utilizing only a CPU).

We also took advantage of the popular Microsoft COCO: Common Object in Context dataset (COCO) [18]. The COCO image dataset seeks to elevate simple object detection by placing objects in their natural context and encouraging complex scene understanding. To achieve these goals the COCO dataset provides non-iconic views of common objects in scenes comprised of many objects versus objects in isolation. The creators argue that other data sets often only include common (iconic) vantage points of objects and models trained on these images will struggle to recognize the same objects in other natural contexts. COCO consists of 118,000 training and 5,000 validation images each with 5 descriptions.

8 Final Image Caption Model Integration

For our architecture we used the merge model described in [16]. We were able to outperform the results in this paper (measured in BLEU on Flickr8k) with

our initial architecture evaluation. There are three phases to the merge model shown in Figure 7.

Feature Extractor: For feature extraction we used pre-trained CNN models described in Section 4. We instantiated these pre-trained models and removed the last layer from the model as it is used to predict a classification label in the original architecture. We were not classifying images, but rather using these models to extract the principal features of the input image in the form of a fixed-length vector. For feature extraction we did not fine-tune any of the weights of the CNN models, but leveraged them with their existing learned weights on the ImageNet dataset. This process is known as transfer learning (see Section 5).

Sequence Processor: During the second phase we used the word embedding layer produced by either GloVe or BERT for handling the text input (see Section 3), which was followed by a LSTM RNN (see Section 6).

Decoder: We implemented 50 percent dropout in both our language and image feature extractor to reduce overfitting on the training dataset. Both our feature extractor and sequence processor produced a 256 element vector, which was concatenated together in the decoder phase using an addition operation. This vector was fed to a 256 dense neuron layer and to a final dense layer. The final layer made a single softmax prediction over the entire output vocabulary for the next word in the sequence. This is known as greedy search.

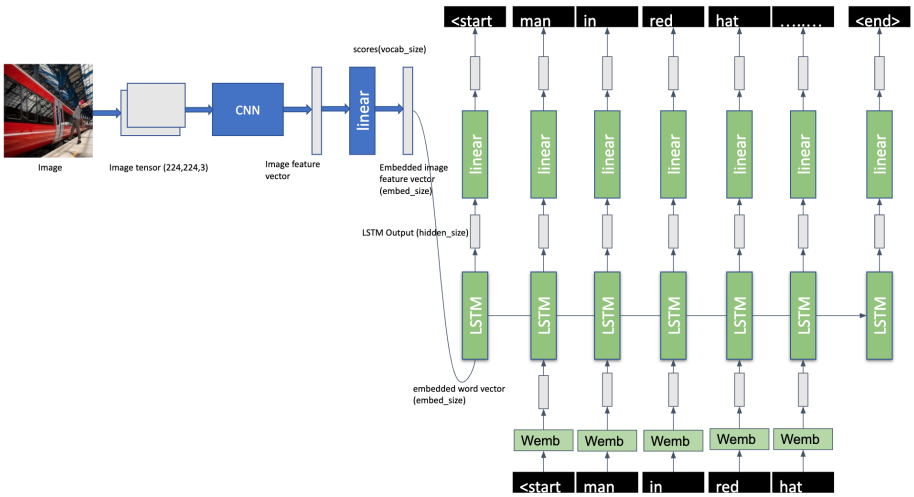

Fig. 7. Encoder/Decoder Architecture

8.1 Data Preparation and Compute Constraints

While our datasets consisted of images with five captions per image, our merge model [16] used much more training data. We prepared our data to combine the features extracted from the image and each sequence data to estimate the next word in the sequence. We formed this data progressively. For example, the token at sequence position one (which happens to be a start token in language generation models) was combined with the features extracted from the image to estimate the token at sequence position two. Next, the extracted image features were combined with the tokens at sequence one and two to estimate the token at sequence position three. This method was applied for each sequence position in each of the captions. This process resulted in training data that was 20 GB in size for Flickr8k and 225 GB in size for COCO.

We ran our models in memory, but recommend that future work use a generator for real time training data preparation (especially for the COCO dataset). We also noted the hurdle of compute for datasets of this size. We ran our Flickr8k models on Google Colab (25GB RAM, 1 vCPU, Tesla K80 GPU) and training took between 8 and 10 hours to process each architecture configuration. Our COCO models ran on a highly optimized Google Cloud Compute instance (600GB RAM, 24 vCPU, Tesla K80 GPU) and took between 18 and 20 hours to process each architecture configuration.

9 Results and Discussion

We implemented a deep encoder-decoder neural network architecture that utilized four different pre-trained CNN models (VGG-16, VGG-19, Resnet, Inception) and two different pre-trained word embeddings (GloVe, BERT). We performed experiments on two datasets, which produced 16 sets of results. These results are shown in Figure 8. For precise BLEU scores please reference Table 1-4 in the Appendix (Section 12).

9.1 Testing and Evaluation

Caption generation models are very difficult to evaluate as syntactic nuance in language is difficult to quantify without human intervention. A popular approach to evaluating description generation was proposed as the Bilingual Evaluation Understudy Score (BLEU) [19]. N-grams are defined as an n-length sequence of contiguous words sampled from text or speech. The BLEU metric is generated by comparing n-gram candidates in our predicted image captions with n-grams in the actual image captions. These n-gram sequences are position independent and in general the more matches that are made between the candidate and reference captions the higher the BLEU score (a value between 0 and 1). The BLEU metric also takes into account that machine translation and image caption generation systems tend to over generate reasonable words. The BLEU metric is designed to not reward candidate descriptions simply by the presence of reasonable words -

what the authors [19] call modified n-gram precision. Our testing metrics utilized unigram, bigram, trigram, and 4-gram BLEU scores for each image averaged across each testing dataset. Intuitively, the larger the n-gram the harder it is to generate a match, which is clear upon quick inspection of our results in Figure 8.

9.2 Results

Through the analysis of our results for the Flickr8k dataset we were able to quickly validate our assumptions that the BERT embeddings would perform better than GloVe. This can be observed in the upper results of Figure 8 by noticing that the scores for each of the models are generally improved when implementing BERT versus GloVe. The Resnet feature extractor performed the best for each of the word embedding schemes, but clearly performed better using BERT. VGG-16 exhibits the next best scores and is also better when using BERT versus GloVe.

The COCO results in the lower results shown in Figure 8 revealed a much tighter distribution when compared with the Flickr8k results. Surprisingly, the BERT embeddings did not improve the BLEU scores for most of the models, with the exception being VGG-16. However, the improvements for utilizing BERT with VGG-16 were not very significant. The Resnet feature extractor also proved to be the best across BERT and GloVe embeddings like observed with the Flickr8k dataset. We were surprised that our models did not perform as well on the COCO dataset in general. Both datasets have sufficiently high resolution images with 5 captions per image and should be comparable. Our hypothesis is that COCO is truly designed to have more objects in context than Flickr8k, which makes the task we tried to perform even harder on this dataset. This discovery was also noted as an opportunity to directly evaluate in future work.

9.3 Application and Future Work

The application of our work is our recommendation that a combination of Resnet image feature extraction and BERT word embeddings are chosen for image captioning tasks when compared with the other architectures evaluated in this paper. This recommendation is valuable considering the cost of training the models in our research. It is notable that the performance gains from our recommendation comes with a cost as we opted to fine-tune our BERT embedding. Fine-tuning BERT embeddings for our specific task was very significant, especially when compared to GloVe embeddings, which we used in their pre-trained state.

There are several areas where improvements can be made for future work. Most of these considerations require additional time and/or compute to be able to quantify improvements. We could explore a wider range of hyperparameters to adjust the number of epochs, dropout rate, embedding size, LSTM output size, LSTM layers, batch size and learning rate of our networks. Next, we could implement beam search instead of greedy search. Greedy search inputs the word with the highest probability at each decoder step, while beam search considers a

set of words at each step. The next word is generated from each word in the set and evaluated with conditional probability. Beam search adds another element of sophistication, but numerous copies of the encoder-decoder network must be applied to every step. Next, we should also consider comparisons between alternatives to handling our word embeddings, such as fine-tuning GloVe embeddings or freezing BERT embeddings in their original state (discussed in Section 5). Finally, we could configure the model architecture to use an attention mechanism as described in Section 2. Attention based models tend to yield better results by allowing the model to attend to information earlier in recurrent layers at the expense of model complexity.

We also discovered many limitations for using BLEU as a metric for image caption generation especially considering our focus was to ensure the correct reference of adjectives and adverbs [20]. Primarily, BLEU scores are position independent, which meant matching n-grams could have completely different ordering and still contribute to a valid image caption score [21]. This also meant that completely nonsensical captions that were not grammatically correct (missing a verb or a reference object) could also contribute to a valid score. Next, we discovered that averaging individual caption scores across a whole corpus will artificially inflate the total score. Furthermore, the authors of [22] found that BLEU negatively correlates with simplicity by rewarding simple language, which is something the creators hoped to prevent. The authors of [23] suggested that even more sophisticated automatic evaluation systems than BLEU are poor at distinguishing outputs of medium and good quality. Finally, the authors [24] found a wide range of correlation between BLEU and human evaluation for similar tasks.

Our future work should consider an evaluation metric that focuses on measuring syntactic features proposed by [25]. These approaches break sentences down into noun phrases, verb phrases, and prepositional phrases as part of the candidate to reference comparison. This would both ensure that adjectives and adverbs are referring to the correct objects while also penalizing sentences that are not grammatically sound. These evaluation discoveries may have impacted the accuracy of our individual results, however we believe that our work and recommendations are still valid from an architecture comparison perspective.

10 Ethics

The promise of harnessing deep learning and transfer learning for image analysis is exciting and full of potential, however, there are important ethical issues that arise. In this section we explore significant hidden subtleties related to bias in datasets, privacy, and misinformation.

10.1 Training Dataset Bias

Deep learning is driven by the availability of large datasets for model training. However, the use of pre-trained models for transfer learning depends on reusing

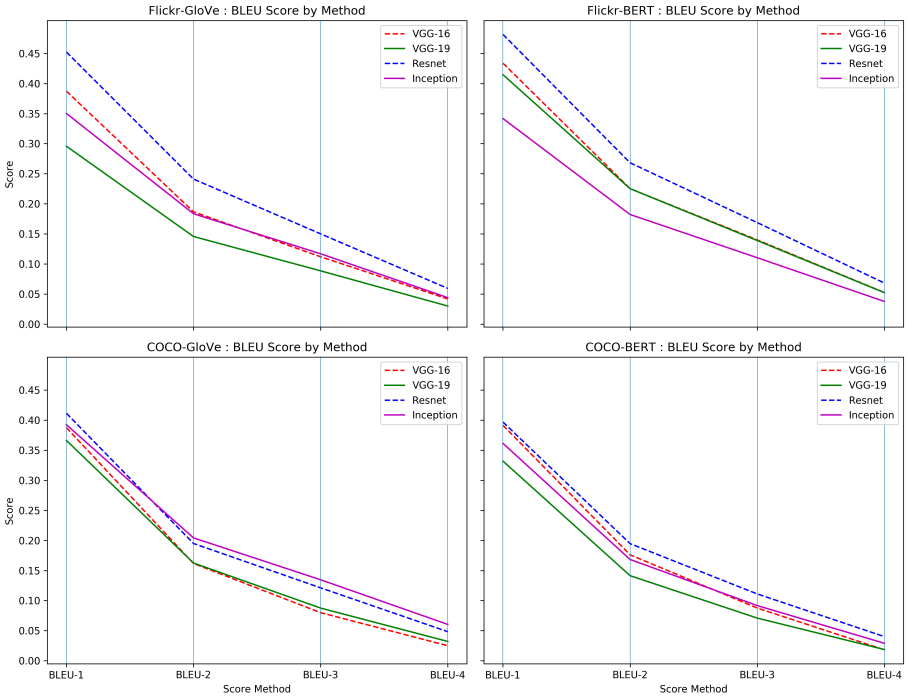

Fig. 8. BLEU Results by Architecture Combinations

others work. This means that anyone putting transfer learning into practice needs to be aware of the potential for bias in these models and take ownership of mitigating or documenting them. We are far from having the capability to produce bias-free models. Professor Vicente Ordóñez [26] describes how he noticed gender bias in the image recognition software he was building and how Microsoft's COCO dataset had significant gender bias. Data biases could also cause harm within other types of image data. For example, cancer lesion detection models built using medical image samples from people in North America will not work well for people in Asia and Africa. Self-driving car models trained on North American roads may struggle in parts of Asia and Africa. This could mean that life-saving cancer diagnosis software will be unavailable to certain parts of the world, or that the safety and efficiencies provided by autonomous vehicles won't be available to less developed countries.

10.2 Privacy

To counteract the obvious harm that can be caused by surveillance using facial recognition a joint effort between policymakers and data practitioners needs to

be created that respects privacy standards. Such policy, should meet the expectations of the public in terms of scope and fairness. Recognizing the lack of consent of the public in general surveillance, the City of San Francisco in May 2019 became the first US city to ban public use of facial recognition [27]. However, the Chinese, Indian and UK governments [28] are moving in the opposite direction with regard to views on privacy in public image recognition.

10.3 Misinformation

Deep learning can now be used for image generation and natural language generation. This is great news for creativity and entertainment purposes, but it can also be used intentionally or accidentally to confuse or mislead people, or outright manufacture facts. Images or videos created using the image generation approaches are known as "deepfakes"[29]. In a similar vein, OpenAI created GPT2 [30] that can generate coherent paragraphs of text. These advancements in deep learning can cause significant harm through misinformation. The solution will be a combination of technical and societal solutions where we track as to where the information originated from, how it is made, and how it is gotten and conveyed by people.

11 Conclusion

Our goal was to improve natural language generation given an image input through the implementation of an image captioning model. Specifically, we set out to improve syntactic relationships between descriptors and their objects. To achieve our goal we presented a series of encoder/decoder merge models utilizing combinations of pre-trained image feature extractors and word embedding schemes through transfer learning. We highlighted the differences in the CNN architectures and our approach to applying word embeddings. We measured the effectiveness of 8 models on 2 datasets, which produced 16 sets of results. We utilized BLEU scores for model comparison, but ultimately found the score to not be effective for our problem statement. We outlined the complexities of working with models and datasets of this scale, while also providing our recommendation on the highest performing architectures. Finally, we highlighted the value and application of our work and presented several compelling opportunities for future work.

References

1. Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *CoRR*, abs/1411.4555, 2014.
2. Kelvin 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. *CoRR*, abs/1502.03044, 2015.

3. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. 2014.
4. Francois Chollet. *Deep Learning with Python*. Manning Publications Co., Greenwich, CT, USA, 1st edition, 2017.
5. Tensorflow core - word embeddings. https://www.tensorflow.org/tutorials/text/word_embeddings. Accessed: 2019-11-04.
6. Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. 2013.
7. Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. 2014.
8. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.
9. Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.
10. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv 1409.1556*, 09 2014.
11. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.
12. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. *CoRR*, abs/1603.05027, 2016.
13. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.
14. Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.
15. Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? pages 3320–3328, 2014.
16. Marc Tanti, Albert Gatt, and Kenneth P. Camilleri. Where to put the image in an image caption generator. *CoRR*, abs/1703.09137, 2017.
17. Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. LSTM: A search space odyssey. *IEEE Trans. Neural Netw. Learning Syst.*, 28(10):2222–2232, 2017.
18. Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.
19. Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A method for automatic evaluation of machine translation. pages 311–318, 2002.
20. Rachel Tatman. Evaluating text output in nlp: Bleu at your own risk. https://towardsdatascience.com/evaluating-text-output-in-nlp-bleu-at-your-own-risk-e8609665a213.
21. Pushpak Bhattacharyya, Sasikumar Mukundan, and Ritesh Shah. Some issues in automatic evaluation of english-hindi mt: more blues for bleu. 01 2007.
22. Elior Sulem, Omri Abend, and Ari Rappoport. BLEU is not suitable for the evaluation of text simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 738–744, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

23. Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.
24. Ehud Reiter. A structured review of the validity of BLEU. *Computational Linguistics*, 44(3):393–401, September 2018.
25. Ding Liu and Daniel Gildea. Syntactic features for evaluation of machine translation. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 25–32, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.
26. Tom Simonite. Machines taught by photos learn a sexist view of women. 2017. https://www.wired.com/story/machines-taught-by-photos-learn-a-sexist-view-of-women/.
27. Kate Conger. San francisco bans facial recognition technology. https://www.nytimes.com/2019/05/14/us/facial-recognition-ban-san-francisco.html.
28. Ravie Laskhmanan. China's new 500-megapixel 'super camera' can instantly recognize you in a crowd.
29. Joseph Foley. 8 deepfake examples that terrified the internet. https://www.creativebloq.com/features/deepfake-examples.
30. Fabienne Lang. Openai's gpt2 now writes scientific paper abstracts. https://interestingengineering.com/openais-gpt2-now-writes-scientific-paper-abstracts.

12 Appendix

Table 1. Results for Flickr8K Dataset with GloVe Embeddings

GloVe	**BLEU-4**	**BLEU**-3	**BLEU-2**	**BLEU**-1
VGG-16	0.041770	0.111938	0.186633	0.387293
VGG-19	0.030013	0.088527	0.145646	0.295786
Resnet	0.059307	0.150023	0.241076	0.452356
Inception	0.043951	0.116926	0.183449	0.350313

Table 2. Results for Flickr8K Dataset with BERT Embeddings

BERT	**BLEU-4**	**BLEU**-3	**BLEU-2**	**BLEU**-1
VGG-16	0.052553	0.139985	0.225280	0.433748
VGG-19	0.052292	0.138914	0.225108	0.414702
Resnet	0.068185	0.168761	0.268223	0.481651
Inception	0.037636	0.110341	0.182212	0.341801

Table 3. Results for COCO Dataset with GloVe Embeddings

GloVe	**BLEU-4**	**BLEU**-3	**BLEU-2**	**BLEU**-1
VGG-16	0.024933	0.079887	0.161678	0.387500
VGG-19	0.032032	0.087531	0.162469	0.366374
Resnet	0.048284	0.121182	0.194929	0.411474
Inception	0.060376	0.134606	0.204241	0.392499

Table 4. Results for COCO Dataset with BERT Embeddings

BERT	**BLEU-4**	**BLEU**-3	**BLEU-2**	**BLEU**-1
VGG-16	0.018199	0.087608	0.176105	0.391411
VGG-19	0.018602	0.018602	0.141391	0.331898
Resnet	0.040080	0.111050	0.194672	0.396902
Inception	0.029009	0.091710	0.168333	0.361405