Image Captioning Using Transformers

Brendan Reidy
bcreidy@email.sc.edu
University of South Carolina
Columbia, South Carolina, USA

a baseball player is about to swing at a pitch

a lot of cars that are on a street

a computer desk with a keyboard and a monitor

Figure 1: Image Captioning Results For Various Images

ABSTRACT

Understanding images/visual input can be difficult for those with visual impairment. Currently, companies like Apple, Google, and Microsoft have ways to annotate images for the visually impaired but these methods have limited scope (training data is sparse for image captioning). Describing images is useful for anyone with visual impairments. Not only is the accuracy of the model important but for real world applications the robustness of the model is also an important factor.

KEYWORDS

datasets, neural networks, image captioning, transformers, natural language understanding

ACM Reference Format:
Brendan Reidy. 2022. Image Captioning Using Transformers. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

1.1 Problem

For the visually impaired the everyday task of understanding scenes, images, or other visual input is challenging. A new architecture of nerual network model known as Transformer models [13] which use multi-headed attention mechanism to understand underlying relationships between words have pushed the boundaries of what is possible for NLP tasks. Models such as BERT [9] and GPT3 [6] have shown massive success in a wide domain of language tasks. This success in the natural language domain has lead some researchers to investigate whether these models can be used for vision related tasks. Herein we deploy the an Encoder-Decoder based transformer model introduced in [13] and evaluate it's performance on image captioning. Furthermore we investigate the robusticity of our models using noise-based adversarial attacks and evaluate the models performance. We deploy our model as a chatbot on a web-based social media platform, Discord, to allow for near real-time image captioning using mobile phones (the bot responds to images with the predicted caption).

1.2 Related Work

Similar works such as Image captioning with deep bidirectional LSTMs [14] use Long Short Term Memory (LSTM) based models to generate captions for images. LSTM's are sequential in nature in that they can only process one word at a time. In their paper, the authors use a bi-directional LSTM which allows the model to pay attention to previous words as well as future words before making a prediction about the word. This allows the model to generalize about the left and right context of words before making a prediction. In their paper the authors achieve a BLEU4 score of 24.4 which was state of the art (SOTA) in 2016 when the paper was released.

Later works use Transformer models in order to generate image captions. The paper Meshed-Memory Transformer for Image Captioning [7] uses a transformer model with memory in order to generate captions. The core idea is that images require multi-modal awareness in order to keep track of which parts of the image have already been described, and which parts still need to be described (e.g. if there are two agents in an image, the caption might describe one agent, and then the other). Using their methodology the authors achieved a new SOTA BLEU4 score of 39.1

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 APPROACH

2.1 Background

We deploy an Encoder-Decoder Transformer model as introduced in the paper Attention is All You Need [13] for the task of image classification. The Transformer model uses an Encoder to "Encode" an input sentence and a "Decoder" to generate new sentences. This structure can be generalized for a wide range of NLP tasks including natural language translation, sequence to sequence text generation, question answering, etc. In order to understand how this works we must look deeper into what each part of the model is doing.

Before data can enter the model, it must be turned into a numeric representation. Text can be turned into numbers using something known as a "Tokenizer". A Tokenizer looks at a corpus of text and assigns each unique word a number. For our purpose, our corpus of text is all the captions in the dataset. Since the number of unique words can be very large, we limit the vocabulary size to $10,000$. Words not in our vocabulary will all be replaced with a number associated with the *unknown* token.

Now that we have a numeric representation of our data, we can use it in machine learning models. However, in its current form the input does not contain very much useful information since we arbitrarily assigned numbers to the words. In order to give words "meaning" they are passed to an embedding layer. The embedding layers job is to out put an n dimensional array for each input token (word). The key idea is that words that are similar in meaning will be close together in terms of euclidean distance on the embedding dimension.

Now we have an $m \times n$ input matrix where m is the length of the sentence an n is the embedding dimension. However, this matrix does not contain any information about the position of a word in a sentence. In order to fix this sinusoidal positional encoding is used (although some models like BERT use a second embedding layer for positional encoding). The key idea behind sinusoidal encoding is that each position has unique values depending on its position in the sentence.

Next, these values are passed to the Multi-headed attention block where scaled dot product attention occurs. The key idea behind scaled dot product attention is that each word is given a score based on how it relates to other words in the sentence. Words that are closely related are given a high attention score. We use multiple attention heads because each attention head can be describing a different type of relation. Using more attention heads allows the model to capture more relations between words.

In the encoder, these values are passed to a feed forward layer, and these values are passed to the next encoder layer if there are more, or they are passed to the decoder layer if it is the last encoder layer. In the decoder, these values are passed to a second multi-headed attention block called the "Encoder-Decoder" attention block. The idea behind this block is to capture associations between the words in the Encoder sentence(s) and the words in the Decoder sentence(s). These values are passed to a feed forward layer, and then to the next decoder layer if there are more, or they are passed to the classification head if this is the last decoder layer. Finally, the classification head is used to make a prediction for the next word in the sentence. The classification head is a single dense layer with *vocab_size* neurons. The largest value is taken to be the output of the model

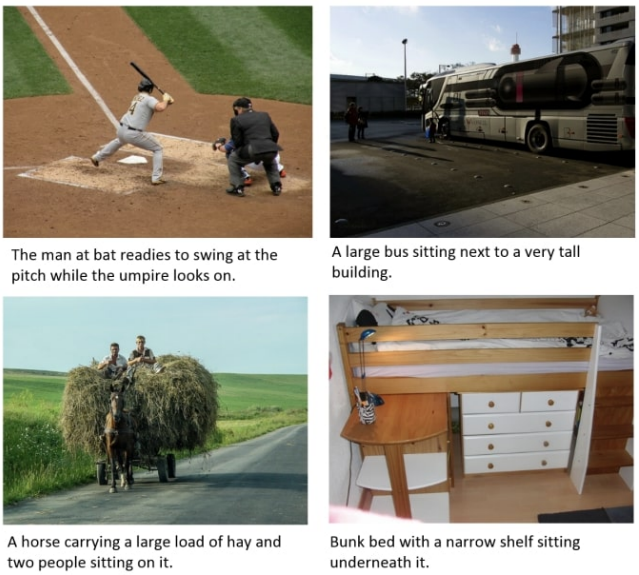

Figure 2: Examples from the MSCOCO Dataset

2.2 Image Transformer

As stated in the previous section the input of the transformer model is text. In order to re-purpose the architecture to work for images, several changes need to be made. First, we replace the embedding layer with the InceptionV3 [12] model which as trained on the ImageNet [8] dataset. Instead of using the entire model, we use the model up to the last layer. This way we generate embeddings for each image using a very powerful pretrained image model. The idea is that this model will capture key information about the images and summarize it using the embedding layer.

Next, we replace the multi-headed attention block in the encoder with a different type of visual attention called Bahdanau attention [4]. The key idea behind Bahdanau attention is that given an image, Bahdanau will highlight areas of interest in an image. These areas of interest are a learned parameter. This reduces the complexity of the problem by allowing the feed forward parts of the model to ignore parts of the images that are not useful.

Finally, the output of the Bahdanau attention is encoded into a single vector via a feedfoward layer so it can be passed to the encoder-decoder attention block.

2.3 Training

For training we use the Microsoft Common Objects in Context [11] dataset or MSCOCO for short. The MSCOCO dataset contains 328,000 images each with 5 captions describing the contents of the image. Figure 2 shows examples of training images for MSCOCO. In order to speed up training time, we generate the image embeddings ahead of time and save them back to the file. We implement our model using custom TensorFlow [3] Keras layers. We use a scheduled learning rate starting at 0.1 with exponential decay, the

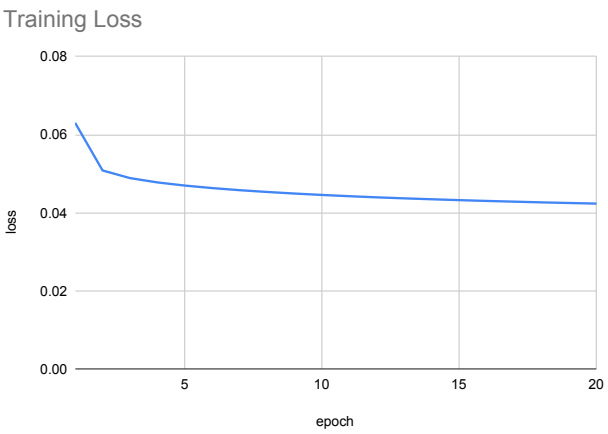

Figure 3: Training loss over the course of 20 epochs

Adam [10] optimizer, and sparse categorical cross-entropy as our loss function. We train our model for 20 epochs. The training loss at each epoch can be seen in figure 3.

3 EVALUATION

For evaluation we use the Bilingual Evaluation Understudy or BLEU score. The BLEU score was designed to evaluate the performance of natural language translation models (e.g. French to English). The evaluation compares a prediction sentence to a group of candidate sentences in order to generate a metric between 0 and 1 that captures the quality of the translation. Although the BLEU score was designed originally for translation tasks, the metric can be used to evaluate any computer generated sentence to a group of candidate sentences e.g. ground truth sentences. We calculate the BLEU score using the nltk [5] python library.

The BLEU score is made of two parts. The first part is modified n-gram precision. N-gram precision measures two things: adequacy and fluency. Adequacy is measured by seeing how many of the same words, or n-gram combos of words, appear in both the predicted and ground truth sentences. Fluency is measured by seeing how long (how many words or n-gram combos) in a row match between predicted and ground truth sentences. The second part of the BLEU score is the Brevity Penalty. The Brevity Penalty punishes sentences that are too long or too short. For our work we measure 1-gram, 2-gram, 3-gram, and 4-gram BLEU score (also known as BLEU1-BLEU4).

4 DISCUSSION

After training the model for 20 epochs we have a Transformer based image captioning model with a BLEU4 score of 12.8. Figure 4 shows our model compared to a baseline model. The baseline model in this case is an identical model, but with no training performed. As seen in the figure, our model out performs the baseline model for all four metrics.

We test the robustness of our model using noise attacks where we replace a percentage of the image with noise and evaluate the output of the model. Figure 5 shows an example of an image that

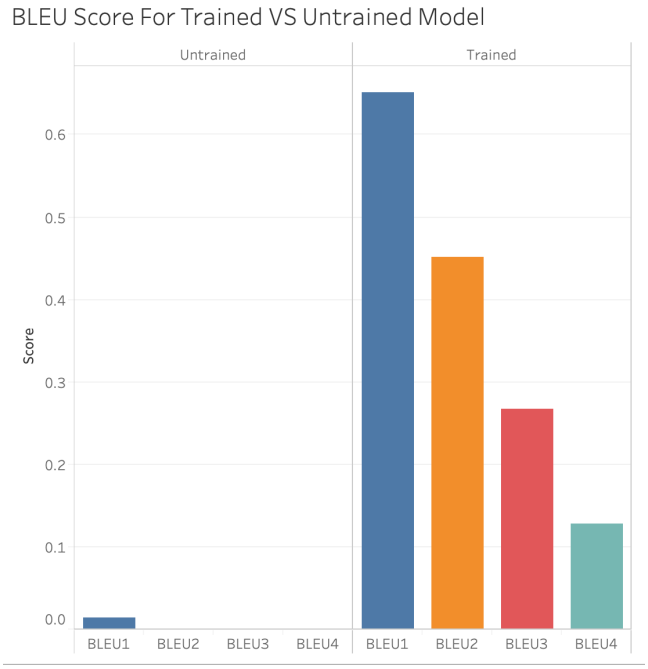

Figure 4: BLEU score comparison with baseline model

Figure 5: Example of noise injected to image. Percentage of noise and generated caption are written under the image.

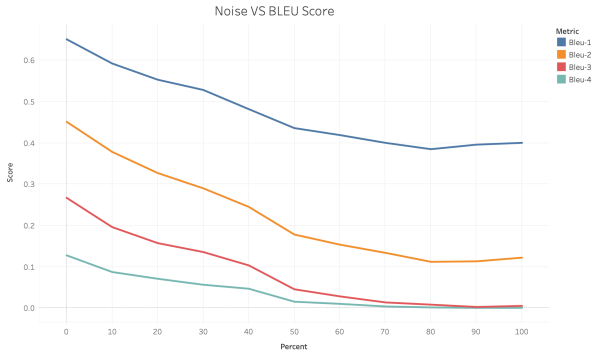

Figure 6: Noise percentage (x axis) vs BLEU score (y-axis).

has varying levels of noise added to it and the output of the model. As shown in the figure, the captions are reasonable for the noise values of 0% and 25%, but the captions quickly change to none-sense for noise values of 50% and 75%. We formalize this observation by evaluating the BLEU score for varying levels of noise injected to the image. Figue 6 shows the effect of noise on the BLEU1 - BLEU4 scores. As we can see, the scores trend downward as the noise level increases. Interestingly, there seems to be a sharp decline in BLEU score for all 4 metrics between the noise percentages of 40% and 50%; indicating that 40% noise is a critical threshold for noise. The BLEU1-BLEU2 scores do not drop to 0 and remain fairly stable after the 50% noise mark. This is likely due to the fact that the model has been trained to produce common words more often, so when evaluating 1-gram or 2-gram combos, the model is able to generate words that are common thus increasing it's BLEU1 and BLEU2 scores. But as the n-gram evaluation increases (e.g. BLEU3 and BLEU4) the likely-hood that the model produces 3 or 4 correct words in a row by random chance decreases sharply.

Figure 1 shows some real world examples of input given to the model and the corresponding output. We connect our model to the Python Discord [2] to allow for near real-time inference using a smart phone camera. Using this set up we use Google Text to Speech API [1] to translate the captions to speech.

5 CONCLUSION

In this paper we demonstrated image captioning using Transformer based computer vision models. We show that our results beat the baseline model maintain some robusticity against noise-based adversarial attacks. We show practical applications of this technology and a direct path to deployment for near real-time image captioning.

Future works could add further improvement for our model by using transfer learning. Using transfer learning a pretrained decoder could be used. This would allow the model to have an underlying understanding of things like grammar, punctuation, fluency, etc before training on the image caption dataset.

REFERENCES

[1] 2022. *Google Text to Speech.* https://pypi.org/project/gTTS/
[2] 2022. *Python Discord API.* https://discordpy.readthedocs.io/en/stable/api.html
[3] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/ Software available from tensorflow.org.
[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. https://doi.org/10.48550/ARXIV.1409.0473
[5] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".
[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. https://doi.org/10.48550/ARXIV.2005.14165
[7] Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. 2020. Meshed-memory transformer for image captioning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 10578–10587.
[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition.* Ieee, 248–255.
[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. https://doi.org/10.48550/ARXIV.1810.04805
[10] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. https://doi.org/10.48550/ARXIV.1412.6980
[11] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. 2014. Microsoft COCO: Common Objects in Context. https://doi.org/10.48550/ARXIV.1405.0312
[12] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the Inception Architecture for Computer Vision. https://doi.org/10.48550/ARXIV.1512.00567
[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. https://doi.org/10.48550/ARXIV.1706.03762
[14] Cheng Wang, Haojin Yang, Christian Bartz, and Christoph Meinel. 2016. Image captioning with deep bidirectional LSTMs. In *Proceedings of the 24th ACM international conference on Multimedia.* 988–997.