Show, Attend and Tell with Transformers

Group ID: 32

Abstract

Image captioning is a now well-established challenge in the Computer Vision/Natural Language Processing community, which consists in generating an accurate description, or caption, of a given input image. This is typically done using a Convolutional Neural Network (CNN) to extract features from the image followed by a language model to sequentially predict an output sentence from these features. This work aims to investigate if by simply using the current state-of-the-art architectures for feature extraction and sequence processing, we can train an accurate image captioning network with a limited amount of resources. We also investigate if incorporating the word embeddings and bounding boxes of objects detected in the image can further improve the model's accuracy. The models are evaluated on the Flickr8k and Flickr30k datasets using the BLEU metric.

1 Introduction

Automatically generating captions from an image is an extremely complicated task, as it requires to jointly solve two problems: detecting objects in an image and establishing a semantic relationship between them, and then expressing that relationship in a linguistically correct way.

Another challenge is defining what a good caption for an image is: we could ask multiple people to describe the same image and end up with no identical answers. This is why image captioning datasets often contain multiple captions for each image. Still, it is not straightforward to tell an algorithm which sentence it should learn to predict, as well as how to evaluate its predictions. As a result, multiple metrics for image captioning evaluation have been developed, such as BLEU (Papineni et al., 2002), CIDEr (Vedantam et al., 2014), METEOR (Lavie and Agarwal, 2007) or SPICE (Anderson et al., 2016).

Hand-designed image feature extraction algorithms like HOG (Dalal and Triggs, 2005) and SIFT (Lindeberg, 2012) have recently lost their popularity to CNNs due to the numerous advances in Deep Learning, which have become the de-facto standard provided enough data is available. CNNs trained on large datasets learn hierarchical representation of image features and are often used to extract useful information into lower dimensions.

Sequence processing has also greatly improved with Deep Learning. Recurrent Neural Networks first demonstrated that they could successfully carry information across multiple time-steps across many NLP tasks. Then, the attention mechanism and the Transformer architecture brought huge improvements to both the Computer Vision and NLP fields. It showed that for any kind of sequential decoding task, allowing an algorithm to "give attention" to specific previous inputs for each prediction could drastically improve its performance.

Since image captioning represents both a vision and language challenge, it naturally has been tackled by trying to come up with methods that take the best of both worlds and combining them together.

2 Related Work

Perhaps the most notable advance in image captioning is Show, Attend and Tell (Xu et al., 2015), a model which utilized a CNN (VGGnet) to extract feature representations from images, and decoded them by employing attention over the images using a LSTM. Anderson et al. 2018 highlighted a flaw in this model and its successors, explaining that attention is computed over a uniform grid of regions, which hinders the models ability to perform attention over significant events occurring in the image. They thus proposed a model that exploits salient image regions for attention (by utilizing the Faster R-CNN object detection model). Desai and Johnson 2021 proposed a pretraining based approach named VirTex, finding that the convolutional feature extraction components of such models can be transferred to other image recognition tasks such as classification and instance segmentation. Additionally, Sariyildiz et al. 2020 proposed ICMLM, a model that learns visual representations of images by predicting masked tokens within their captions, further cementing the notion that transformer architectures have high capability. Alternatively, Zhang et al. 2020 demonstrated that utilizing bidirectional contrastive objectives between text and image modalities yield high performing models for learning image representations. Radford et al. 2021 trained a simpler variant of this using 400M image-text pairs, demonstrating its efficiency and zero-shot capability.

Given the abundance of noisy textual data in the web, weak supervision in image captioning has been an interesting avenue for research. By training CNNs using the YFCC100M (Thomee et al., 2016) dataset and a novel loss function based on Jelinek-Mercer smoothing, Li et al. 2017 proposed a model that generates relevant n-grams from images. Additionally, Mahajan et al. 2018 and Yalniz et al. 2019 considered weak supervision on datasets consisting of 3.5 and 1 billion Instagram images respectively, both yielding models with good accuracy.

With the release of bigger and bigger datasets, most of the recent methods have been training huge models end-to-end, but this is not always possible when the computational budget is restricted and sometimes not even necessary. The aim of this work is therefore to determine if lighter models and methods can still achieve convincing results.

3 Methods

In this section we propose a model for image captioning combining a pre-trained CNN image classifier with a modified transformer architecture. In particular, our approach consists of the following:

1. The input image is fed to the image classifier and the image features are extracted.
2. The extracted features are passed through a linear layer, used for dimensionality reduction.
3. The reduced features are fed to our modified transformer architecture and the caption is produced.

3.1 Image features extraction and dimensionality reduction

In this work, we use EfficientNet (Tan and Le, 2019), to extract spatial image features. EfficientNets are a family of CNN image classifier models that achieve state-of-the-art accuracy with improved efficiency up to an order of magnitude (i.e. fewer parameters). There are 7 different versions released, each with increasing number of parameters, inference time, but also accuracy. As a trade-off between these, we decided to use EfficientNet-v4 in all our experiments.

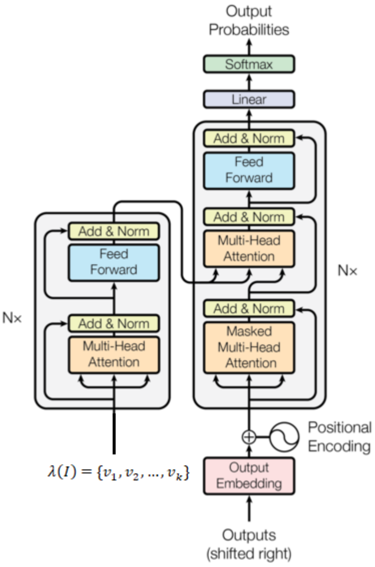

Figure 1: Modified transformer

Firstly, the input image I of the size $3\times224\times224$ is passed to the pre-trained EfficientNet. The required image features are simply taken to be the output of the final convolution layer of the model. This procedure yields a feature matrix of size $1792 \times 7 \times 7$. The modified features are then transformed into dim_{model}-dimensional vectors via a linear layer for dimensionality reduction purposes. To achieve this, we firstly flatten the last two dimensions of the feature matrix to obtain matrix of the size 1792×49. Then, we feed the new matrix to the simple feedforward layer to obtain a matrix of the size $dim_{model} \times 49$. Therefore, for each image I the previous procedure produces the fixed size (i.e. 49 elements) set of dim_{model}-dimensional vectors, which we denote by:

$$EN(I) = \{v_1, v_2, ..., v_{49}\}.$$

3.2 Modified transformer

Our modified transformer is based on the original Transformer architecture introduced by Vaswani et al. 2017. However, unlike the original architecture that takes a sequence of words as an input, our model takes features extracted from images, which influences our design choices.

Model	Object detector	GloVe	Heads	Embedding dim	Feedforward dim	Dataset
1	No	No	4	64	256	Flickr8k
2	No	Yes	2	50	256	Flickr8k
3	Yes	No	2	54	256	Flickr8k
4	No	No	4	128	512	Flickr30k
5	No	Yes	5	50	512	Flickr30k

Table 1: Different models trained

Firstly, unlike the original transformer, there is no input embedding layer. Instead, we assume that the (image) features are properly encoded beforehand. Given an input image I, we denote a function returning the set of feature vectors by $\lambda(I)$. Our main model uses $\lambda(I) = EN(I)$.

Another difference is that we do not employ any positional encoding of the set of feature vectors, because they do not have any positional structure relevant for the captioning task.

The rest of the network is the same as original transformer. The full modified transformer is presented in Figure 1. We now recall some basics of the Transformer model as presented in (Vaswani et al., 2017).

Encoder: the encoder part of our model consists of $N = 3$ identical encoding layers. As can be seen from the figure, each encoder layer consists of a multi-head self-attention sub-layer and a feed-forward sub-layer, each with residual connection and layer normalization. Both of these sub-layers produce dim_{model}-dimensional vectors. If X is an input to the Encoder layer, then :

$$Y_1 = LayerNorm(X + MultiHead(X))$$

$$Y = LayerNorm(Y_1 + FeedForward(Y_1))]$$

where Y is the final output of the Encoder layer.

Decoder: the decoder part of our model also consists of $N = 3$ identical decoding layers, each of which contains both the previously mentioned sub-layers and a multi-head attention sub-layer applied to the outputs of the encoder. However for the self-attention, we employ masking to avoid peeking ahead.

Attention: the multi-head attention consists of h heads each corresponding to a separate scaled dot-product attention attending information from different subspaces in parallel.

$$MultiHead(Q, K, V) =$$

$$Concat(head_1, ..., head_h)W^O$$

where

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

and scaled dot-product attention is defined as

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{dim_{model}}})V$$

where Q, K, V are the query, key and value matrices with W^O being a linear transformation combining the attention of the separate heads.

Feed-forward sub-layer: the feed forward layer consists of two linear transformations with a ReLU activation after the first one:

$$FeedForward(x) = ReLU(xW_1 + b_1)W_2 + b_2$$

where W_1, b_1 are the weight matrix and bias of the first linear transformation, and W_2, b_2 are weight matrix and bias of the second linear transformation. Note that the output of $FeedForward$ is dim_{model}-dimensional, but the output of the first linear transformation is $dim_{feedforward}$, both of which are hyperparameters.

Positional encoding of the output: although we do not use positional encoding for the input, we add the following positional encoding to the output embedding:

$$PE_{(pos,2i)} = sin(pos/10000^{2i/dim_{model}})$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/dim_{model}})$$

Output embedding: the two main approaches we employ is either learning the embeddings in the same fashion as (Vaswani et al., 2017), or we use pre-trained GloVe embeddings. The particular choice for each of the models can be found in Table 1. The output sequence is shifted by one position ensuring that the predictions can depend only on the previous outputs.

Objective function: the output of the final layer in the decoder is fed to a linear layer, that outputs

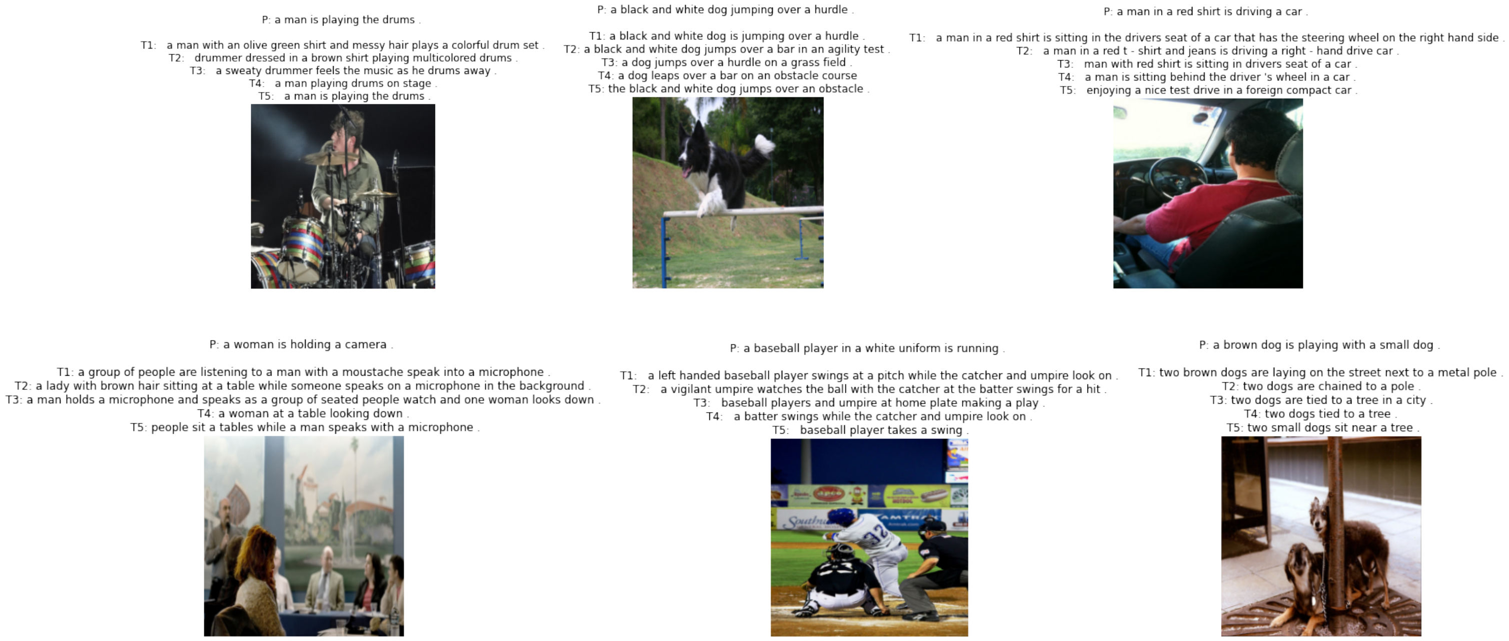
P: a man is playing the drums .
T1: a man with an olive green shirt and messy hair plays a colorful drum set .
T2: drummer dressed in a brown shirt playing multicolored drums .
T3: a sweaty drummer feels the music as he drums away .
T4: a man playing drums on stage .
T5: a man is playing the drums .
P: a black and white dog jumping over a hurdle .
T1: a black and white dog is jumping over a hurdle .
T2: a black and white dog jumps over a bar in an agility test .
T3: a dog jumps over a hurdle on a grass field .
T4: a dog leaps over a bar on an obstacle course
T5: the black and white dog jumps over an obstacle .
P: a man in a red shirt is driving a car .
T1: a man in a red shirt is sitting in the drivers seat of a car that has the steering wheel on the right hand side .
T2: a man in a red t - shirt and jeans is driving a right - hand drive car .
T3: man with red shirt is sitting in drivers seat of a car .
T4: a man is sitting behind the driver 's wheel in a car .
T5: enjoying a nice test drive in a foreign compact car .
P: a woman is holding a camera .
T1: a group of people are listening to a man with a moustache speak into a microphone .
T2: a lady with brown hair sitting at a table while someone speaks on a microphone in the background .
T3: a man holds a microphone and speaks as a group of seated people watch and one woman looks down .
T4: a woman at a table looking down .
T5: people sit a tables while a man speaks with a microphone .
P: a baseball player in a white uniform is running .
T1: a left handed baseball player swings at a pitch while the catcher and umpire look on .
T2: a vigilant umpire watches the ball with the catcher at the batter swings for a hit .
T3: baseball players and umpire at home plate making a play .
T4: a batter swings while the catcher and umpire look on .
T5: baseball player takes a swing .
P: a brown dog is playing with a small dog .
T1: two brown dogs are laying on the street next to a metal pole .
T2: two dogs are chained to a pole .
T3: two dogs are tied to a tree in a city .
T4: two dogs tied to a tree .
T5: two small dogs sit near a tree .

Figure 2: Top: Predictions (P) with targets (T1-T5) on the test set with highest BLEU score. Bottom: Predictions with lowest (= 0) score.

a vector with the dimension being equal to the vocabulary size. The softmax is applied to predict next token probabilities and the loss function we minimize is cross entropy loss, where $y^*_{1:T}$ is a ground truth sentence and y^*_t is the prediction at position t:

$$L(\theta) = -\sum log(p_\theta(y^*_t|y^*_{1:t-1}))$$

3.3 Ablation model

We perform the ablation study by adding more feature vectors to $\lambda(I)$. In particular, we use an object detector on the input image to obtain a set of detected objects as words and we embed those words into $(dim_{model} - 4)$-dimensional vectors. We also use positions of the top-left and bottom-right corners of the bounding boxes of detected objects to create 4-dimensional vectors containing those coordinates. Then, for each of the detected objects the dim_{model}-dimensional representation is produced consisting of 4-dimensional coordinates vector and $(dim_{model} - 4)$-dimensional word embedding. For our particular purpose we use YOLOv5 as object detector and GloVe for word embedding, and we denote the set of obtained dim_{model}-dimensional vectors by $YL(I)$.
We then characterise λ as:

$$\lambda(I) = Concat(EN(I), YL(I))$$

Since the object detector can detect different number of objects in different images, we use 0-vectors of dimension dim_{model} to create suitable padding.

3.4 GloVe embeddings

When the amount of data available is limited, it can prove useful to use models pre-trained on much larger datasets, as they can better capture global and/or local semantic representations of the data and help avoid overfitting. GloVe (Global Vectors for Word Representation, (Pennington et al., 2014)) is a famous word embedding technique, with pre-trained embeddings of different dimensions available online. We use GloVe-50 in our experiments.

3.5 Dataset

We perform experiments on two famous image captioning datasets, Flickr8k and Flickr30k. Flickr8k contains 8000 images, which we randomly split between 6000/1000/1000 train/validation/test images. Flickr30k has 30 000 images, which we split in 28 000 for training, 1000 for validation and testing. For both datasets, each image has 5 corresponding target sentences. During training, we randomly select one for each image in the batch. During evaluation, we compare the model's prediction with all 5 captions using the BLEU metric.

3.6 Bilingual Evaluation Understudy (BLEU)

BLEU (Papineni et al., 2002) is a metric used for comparing the similarity between a candidate sentence and multiple reference sentences. It is computed as:

$$BLEU = P_B \times exp\left(\sum_{n=1}^{N} w_n log(p_n)\right)$$

where P_B is the brevity penalty defined as:

$$P_B = \begin{cases} 1 & \text{if} \quad r > c \\ e^{1-\frac{r}{c}} & \text{if} \quad r \leq c \end{cases}$$

and p_n is the n-gram modified precision score:

$$p_n = \frac{\sum\limits_{C \in \text{Candidates}} \sum\limits_{C \in \text{n-gram}} \text{Count}_{\text{clip}}(\text{n-gram})}{\sum\limits_{C' \in \text{Candidates}} \sum\limits_{C' \in \text{n-gram}'} \text{Count}(\text{n-gram}')}$$

We use BLEU throughout all our experiments as evaluation metric.

3.7 Beam Search

One easy and intuitive way to make predictions at inference time with the trained Transformer is to use Greedy Search: at each time step, we predict the word with highest probability and concatenate it to our current sentence prediction, which we feed to the Transformer as input for the next time step. While very fast, this method does not guarantee to find the most likely sentence under our model's predictions. This could be achieved by an Exhaustive Search, but this is usually computationally intractable and inefficient. Beam Search works in between both methods by only keeping the top-k most likely predictions up to the current time step during decoding, and finally selecting the sequence with the highest cumulative logarithmic score once all the top-k sequences are completed, i.e. they have reached an end-of-sentence token. We use Beam Search with a beam size of 3 when computing the validation BLEU-4 score.

4 Experiments

In our experiments we do the following:
For the Flickr8k dataset, we train 3 networks: a network with and without pre-trained GloVe embeddings (for the transformer's decoder), and a network without pre-trained embeddings but with a pre-trained object detector. The detector returns bounding box coordinates together with the class of detected objects. The detected objects are mapped through the GloVe embedding matrix and then concatenated with their corresponding box coordinates to create 54-dimensional vectors, before being concatenated again with the transformed image features from the classifier. We investigate if these additions facilitate generating good image captions. For the Flickr30k dataset, we only train models with and without GloVe, as using the object detector is too expensive. We study the generalization capabilities and limitations of the best models in the section 5. A summary of the models together with their hyperparameters (number of multi-attention heads, number of neurons in the feedforward layer and the embedding dimension) is given in Table 1.

For all experiments we use the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$ and a batch size of 128.

4.1 Data pre-processing and augmentation

We pre-process the images as follows:
We apply data augmentation techniques on the images to increase the generalization capabilities of our models. For each image, we apply a gamma correction (with γ being randomly chosen among 0.8, 1.0 (no correction) and 1.2), random horizontal flip with 0.5 probability and random cropping followed by resizing, such that each image in the batch always has dimensions 3x224x224. Gamma correction affects the brightness of an image by applying a simple modification to each pixel's intensity:

$$I' = 255 \times \left(\frac{I}{255}\right)^{\gamma}$$

We then apply pixel normalization on each channel of the images following the same normalization used during the training of EfficientNet-v4. At inference time we remove all data augmentation.

Regarding text data: we decapitalize all words in the datasets, and during training we prepend a "start-of-sentence" and append a "end-of-sentence" token to each caption.

4.2 Model Selection

For model selection we use the Cross Entropy loss to train and Early Stopping to detect when the BLEU-4 validation score stops increasing. Then, for each saved model, we evaluate it on the validation set with a beam size from 1 to 5, and we pick our final model as the one with highest validation BLEU-4 score. We then evaluate quantitatively and qualitatively this best model on the test set. This is done for both Flickr datasets.

Dataset	Model	k	BLEU-4
Flickr8k	1	5	**20.225**
Flickr8k	2	5	17.55
Flickr8k	3	4	19.5
Flickr30k	4	5	**19.004**
Flickr30k	5	3	16.75

Table 2: BLEU validation scores with the best beam size k.

Dataset	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Flickr8k	Soft-Attention	67	44.8	29.9	19.5
Flickr8k	**Ours (Model 1)**	**64.14**	**43.88**	**30.19**	**20.565**
Flickr8k	Hard-Attention	67	45.7	31.4	21.3
Flickr30k	**Ours (Model 4)**	**59.9**	**39.36**	**26.1**	**17.35**
Flickr30k	Soft-Attention	66.7	43.4	28.8	19.1
Flickr30k	Hard-Attention	66.9	43.9	29.6	19.9

Table 3: BLEU test scores.

5 Results and Discussion

5.1 Quantitative Results

The test results are compared to those in Xu et al. 2015 and are reported in Table 3.

We can see that by simply using EfficientNet-v4 and a Transformer architecture for Image captioning, we get a test BLEU score of 20.565, which outperforms the Soft-Attention version and is close to matching the accuracy of the Hard-attention model on Flickr8k. We notice a decrease in performance on Flickr30k, where we only reach a 17.35 BLEU score. This is in part due to the random splitting into the different sets, as for Flickr8k we get a better test score than validation, which is not the case for Flickr30k, but also due to the strong bottleneck of BLEU's evaluation capabilities, which we demonstrate later on. Adding GloVe embeddings or the object detector yields worse results, which we explain by the fact that using pre-trained low-dimensional embeddings might restrict the network too much, while having the object detector's inputs leads to overfitting.

It is important to note that due to computational resources constraints, we did not manage to train the models until full convergence: they were all trained for around 100 epochs and all were showing a continuously decreasing loss, but surprisingly most had attained their best BLEU validation score after a small number of epochs (10 to 20). This highlights the limited amount of resources required to obtain such results, which can prove useful in many restricted settings.

We suggest four reasons as to why this behaviour might occur. The first obvious one would be that the network is overfitting: We have a fairly small number of images and the Transformer is a very powerful architecture, so this could be one plausible explanation. However, since even on Flickr30k we very quickly get a good BLEU score which then stays fairly constant for many epochs, our second belief is that it is likely that the model quickly learns the general "shape" of sentences together with global recurring objects in the image. For example, it is very common to find in at least one of the five target sentences tokens like "a", "in the", "in a", as well as objects like "dog", "people", "boy". It is then possible that fine-tuning the model to actually focus further on details of the images takes a lot more training time, which could be indicated by the training loss continuously decreasing in all our experiments.

Also, a potentially negative effect that could be happening is that instead of focusing on predicting better sentences, the model is decreasing the loss simply by becoming even more confident in its correct predictions, which is common in classification scenarios and is undesirable as it does not focus on getting the rest of the descriptions right.

Further study at intermediate stages of training could help identify which one these scenarios, if any, is occurring.

Target	I love playing volleyball	100
Prediction	Sometimes I love playing volleyball	66.87
Prediction	I sometimes love playing volleyball	0
Prediction	I love sometimes playing volleyball	0
Prediction	I love playing sometimes volleyball	0
Prediction	I love playing volleyball sometimes	66.87

Table 4: BLEU-4 limitations: Regardless of the overall meaning of the sentence, one word can dramatically impact the score assigned to a prediction.

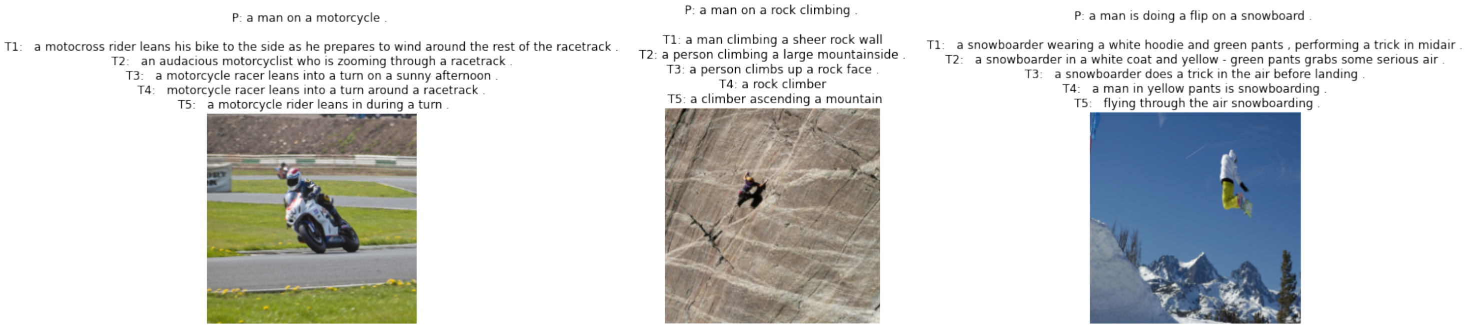
P: a man on a motorcycle .
T1: a motocross rider leans his bike to the side as he prepares to wind around the rest of the racetrack .
T2: an audacious motorcyclist who is zooming through a racetrack .
T3: a motorcycle racer leans into a turn on a sunny afternoon .
T4: motorcycle racer leans into a turn around a racetrack .
T5: a motorcycle rider leans in during a turn .
P: a man on a rock climbing .
T1: a man climbing a sheer rock wall
T2: a person climbing a large mountainside .
T3: a person climbs up a rock face .
T4: a rock climber
T5: a climber ascending a mountain
P: a man is doing a flip on a snowboard .
T1: a snowboarder wearing a white hoodie and green pants , performing a trick in midair .
T2: a snowboarder in a white coat and yellow - green pants grabs some serious air .
T3: a snowboarder does a trick in the air before landing .
T4: a man in yellow pants is snowboarding .
T5: flying through the air snowboarding .

Figure 3: Examples where the model predicts a good caption (P) but gets a BLEU score of 0.

Finally, by inspecting the predictions our trained models make on images taken from the training and validation sets, we conclude that using the BLEU metric together with only 5 target sentences often yields a very poor method to evaluate the predicted captions. This is explored in greater depth in the following section.

5.2 Qualitative Results

We show some of our models' best and worst predictions (in the BLEU-4 sense) in Figure 2. By inspecting some of the supposedly worse predictions, we notice a huge limitation in using the BLEU metric together with the Flickr datasets. The first one, illustrated in Table 4, is that for a difference of only one word between the true and predicted caption, the corresponding BLEU score can vary extremely based on where that extra word is.

This is because BLEU is computed with a geometric mean over precision scores with different n-grams. Therefore, as soon as any of the precision scores is low (for example the 4-grams precision is 0), this brings the BLEU score to basically 0, even though the entire description could be qualified as good. Additionally, because there are only 5 target sentences per image, our model sometimes predicts a correct caption, but because it is not in the 5 reference sentences and due to how BLEU is computed, we can often get a score of 0 for a perfectly valid caption. We show multiple images where this is the case in Figure 3. This clearly outlines how poor BLEU can sometimes be in estimating the correctness of predictions, which is also due in big part to the small amount of target captions per image.

6 Conclusion

In this research, we have investigated if we could attend the image features of EfficientNet-v4 with a Transformer to solve the image captioning problem on two datasets, Flickr8k and Flickr30k. We have shown that this approach works extremely well, and that the main bottleneck we encountered was using the BLEU metric. This metric, when used with a limited amount of target captions, can easily yield a score of 0 for perfectly correct descriptions, which makes evaluation and therefore model selection extremely tedious. A version using pre-trained GloVe embeddings and one using a pre-trained object detector were also studied but did not show to add additional value and instead worsened the results.

For future work, a clear direction is to repeat the same experiment but using different evaluation metrics such as METEOR or SPICE, as well as bigger datasets such as COCO, since our experiments demonstrate that our proposed models were able to learn very good descriptions even on the Flickr30k dataset. One clear advantage of our method is its implementation simplicity and intuitiveness as well as its restricted computational budget, as we were able to obtain convincing results in few epochs by using a simple instance of Google Colab Pro. As explained in section 5, we also consider that studying how the loss' decrease after the first few epochs actually correlate with the change in the network's predictions, to better understand if the bottleneck is overfitting or rather BLEU and the limited amount of target sentences. Finally, we still believe that adding an object detector's predictions as input to the Transformer will prove useful in some scenarios, but this will require detectors able to detect many more objects than what YoloV5 currently achieves (roughly 1 object per image on average).

References

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. SPICE: semantic propositional image caption evaluation. *CoRR*, abs/1607.08822.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086.

N. Dalal and B. Triggs. 2005. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1.

Karan Desai and Justin Johnson. 2021. Virtex: Learning visual representations from textual annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11162–11173.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. pages 228–231.

Ang Li, Allan Jabri, Armand Joulin, and Laurens Van Der Maaten. 2017. Learning visual n-grams from web data. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4183–4192.

Tony Lindeberg. 2012. *Scale Invariant Feature Transform*, volume 7.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. 2018. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pages 181–196.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Mert Bulent Sariyildiz, Julien Perez, and Diane Larlus. 2020. Learning visual representations with caption annotations. In *European Conference on Computer Vision*, pages 153–170. Springer.

Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR.

Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. 2016. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2014. Cider: Consensus-based image description evaluation. *CoRR*, abs/1411.5726.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR.

I Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Mahajan. 2019. Billion-scale semi-supervised learning for image classification. *arXiv preprint arXiv:1905.00546*.

Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D Manning, and Curtis P Langlotz. 2020. Contrastive learning of medical visual representations from paired images and text. *arXiv preprint arXiv:2010.00747*.