Image Caption Generation

Lucia Hainline
School of Engineering and Applied Science
Columbia University
New York City, NY, United States
lph2126@columbia.edu

Anouck Rietveld
School of Engineering and Applied Science
Columbia University
New York City, NY, United States
ar4633@columbia.edu

***Abstract*—This paper draws inspiration from the seminal study "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention". We implement an encoder-decoder framework using a VGG16-based neural network as the encoder and an LSTM-based decoder for generating image captions on the Flickr8k dataset. To evaluate the effectiveness of attention mechanisms, we compare the performance of our attention-based model with a baseline model that does not incorporate attention. Furthermore, we investigate the differences between soft and hard attention approaches, analyzing their impact on model performance. Our results are benchmarked against those reported in the original 'Show, Attend and Tell' study, highlighting similarities and deviations in findings. Metrics such as BLEU and METEOR scores are used to quantify model performance. This work contributes to understanding the practical implications of attention mechanisms in image captioning tasks and provides insights into their strengths and limitations.**

***Index Terms*—image captioning, visual attention, computer vision, natural language processing, deep learning.**

I. Introduction

Image captioning, the task of generating textual descriptions for images, is a challenging problem at the intersection of computer vision and natural language processing. It requires a model to effectively understand the visual content of an image and translate it into coherent, meaningful text. The advent of deep learning has enabled significant advancements in this domain, with encoder-decoder architectures becoming the foundation for state-of-the-art models.

Since 2012, there has been a growing interest in addressing the image captioning problem, driven by advancements in training neural networks. Early approaches utilized convolutional neural networks (CNNs) for image feature extraction in combination with recurrent neural networks (RNNs) for sequence generation. The introduction of attention mechanisms, as highlighted in the "Show, Attend and Tell" study, enabled models to focus on specific regions of an image during caption generation, enhancing descriptive accuracy.

Subsequent research explored various attention-based models, including soft and hard attention, and incorporated object detection techniques to improve caption relevance. More recently, the field has seen a significant shift towards transformer-based architectures and multimodal learning approaches. Originally designed for natural language processing tasks, transformers have been adapted for image captioning due to their ability to model long-range dependencies and handle variable-length sequences. These architectures have propelled the field forward, enabling models to generate more human-like and contextually appropriate captions [2].

Moreover, the integration of multiple modalities—such as text, audio, and visual data—has become a focal point in recent research [3]. Multimodal models aim to learn joint representations that encapsulate information from different sources, enhancing their ability to understand and describe complex scenes. Additionally, the advent of large language models (LLMs) has opened new avenues for image captioning, allowing models to generate descriptive text based on visual inputs when combined with visual encoders [4]. However, despite the significant progress achieved over the years, image captioning remains far from being a solved task [5].

In this work, we draw inspiration from the seminal study "Show, Attend and Tell" to explore the role of attention mechanisms in neural image captioning. Specifically, we implement an encoder-decoder framework where image features are extracted using the VGG16 neural network and captions are generated using a Long Short-Term Memory (LSTM) based decoder. We focus on comparing the performance of models with and without attention mechanisms and further contrast the effectiveness of soft versus hard attention in generating accurate and visually grounded captions. Through qualitative and quantitative evaluations on the Flickr8k dataset, we demonstrate the benefits of attention mechanisms and visualize how attention focuses on relevant regions of the image during caption generation.

II. Summary of the Original Paper

In the paper "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", the authors use the Flickr8, Flickr30, and MS COCO datasets to caption images using two different attention mechanisms. The two mechanisms are used within the same model framework, an encoder-decoder model pre-trained on the Oxford VGGnet model. The two mechanisms of attention are described as "soft" deterministic attention and "hard" stochastic attention. The value of these attention mechanisms in image captioning is that the authors can gain insight into where and what the model focuses on when captioning the images. The paper first summarizes related work, sharing the advances made in image captioning techniques since the use of neural networks. They mention that

most approaches "represent images as a single feature vector from the top layer of a pre-trained convolutional network" [1], which is how they represent images in their model.

After a brief review of the literature, the authors proceed to the details of the model. In each variant, they use an encoder with features extracted from the lower convolutional layers of the pre-trained VGGNet model. This is a novel approach compared to the previous literature, in which the encoder leverages a fully connected layer for output. According to the authors, "this allows the decoder to selectively focus on certain parts of an image by selecting a subset of all feature vectors" [1].

The decoder uses an LSTM network to generate "... one word at every time step conditioned on a context vector, the previous hidden state, and the words generated before" [1]. They describe the context vector as a representation of the relevant part of the image at each time step. The context vector is created using their two attention mechanisms. First, the authors describe the "hard" attention mechanism, which is stochastic. Hard attention only considers isolated locations while determining the next word, and ignores everything else. The mechanism samples a location from a multinoulli distribution, making the context vector a random variable. The parameters of this distribution are derived from the output of the LSTM decoder, which will tell the model where to focus attention in the image. The authors compute the gradient of the objective function using Monte Carlo sampling, with the attention location as a random variable. The gradient is approximated using the sampled location, and we apply a moving average baseline to reduce the variance. The chosen location and its features are used to produce the next word in the predicted caption.

Next is "Soft" Attention, which is deterministic and differentiable, therefore it can be used seamlessly in backpropagation. It assigns a continuous weight to each part of the input, producing a weighted sum of the features where the weights represent the importance of each part. These weights are computed using a dot product, and normalized with a softmax function. Unlike hard attention, which selects a single input part, soft attention distributes focus across all parts. The location with the most attention is used to predict the next word.

Once the encoder/decoder architecture is built, the authors train the model on three data sets: Flickr8, Flickr30, and MS COCO. For Flickr8, they use the RMSProp learning rate and Adam for the larger datasets. In addition, the authors used dropout and early stop during training, and focused on the BLEU scores to interpret the results. For each experiment, the authors use a vocabulary size of 10,000, and they use the provided or publicly available train, test, validation splits from the datasets.

The most successful models use the MS COCO dataset with Hard Attention. They achieved a BLEU-1 score of 71.8 for this model. Across all the datasets, the highest BLEU scores come from the model that uses the Hard Attention mechanism. Interestingly, the models using Flickr8 have higher BLEU scores than those with Flicker30, but MS COCO outperforms these in every experiment.

III. Methodology

In this section, we describe the two variants of our attention-based model, as well as the model without attention. We will begin by outlining their common framework, which diverges in the decoder section.

A. *Encoder: VGG16 Neural Network*

The encoder of our model is based on the VGG16 convolutional neural network, a 16-layer deep architecture pre-trained on the ImageNet dataset, which contains over a million images across 1,000 object categories.

We utilized the pre-trained network without fine-tuning, focusing on feature extraction rather than classification. Specifically, we removed the fully connected layers at the end of the network, as they are tailored for the original classification task. Instead, we extract features from the $14 \times 14 \times 512$ feature map produced by the fifth convolutional layer (prior to the max-pooling operation).

The extracted feature map was flattened into a 2D representation of size 196×512, where $L = 196$ represents the number of spatial locations ($14 \times 14 = 196$) and $D = 512$ is the depth of the feature map, corresponding to the dimensionality of each feature vector. This transformation is illustrated in Figure 1. The resulting flattened representation served as the input to the

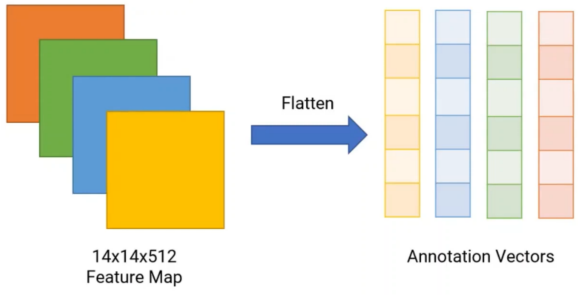

Fig. 1. Flattening of the feature map into annotation vectors. [8]

decoder, as shown in Figure 2. By extracting features from a lower convolutional layer, the decoder can selectively focus on different spatial regions of the image, enabling it to attend to specific areas. Each of the resulting L feature vectors, or also referred to as annotation vectors $\mathbf{a}_i$, provides a D-dimensional representation of a distinct part of the image. This approach aligns with the method described by Xu et al. (2015) [1]. These vectors form the foundation of the attention mechanism employed by the decoder, which we elaborate on in the next section.

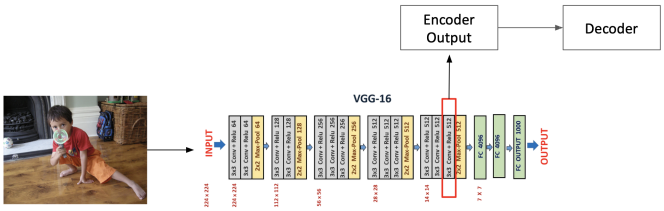

Fig. 2. Pipeline for the encoder with the pre-trained VGG16 Model [7].

B. *Decoder: Long Short Term Memory*

The decoder of the model is an RNN-based LSTM architecture designed to combine image and text processing. The LSTM processes the textual input and sequentially generates captions conditioned on the visual features extracted by the encoder. The decoder integrates the visual features with the textual context at each timestep to produce contextually appropriate captions.

The methodology of the decoders across the three models is centered on generating captions sequentially by integrating visual and textual information. All decoders utilize a Long Short-Term Memory (LSTM) network to process the sequential nature of captions, leveraging its ability to capture dependencies between words. The decoders incorporate an embedding layer to transform word indices into semantic vector representations, which are then combined with visual features extracted by the encoder. These features are processed through dense layers to ensure compatibility between modalities. The final word prediction is made using a dense layer with a softmax activation function, producing a probability distribution over the vocabulary. The LSTM combines the image and text features to learn a joint representation and outputs a probability distribution over the vocabulary for predicting the next word in the caption. While the baseline model directly processes the combined features, the attention based models incorporate mechanisms to focus on relevant regions of the image, refining context and improving caption quality. The application of both the soft and hard attention model is in line with Xu et al. [1] and these additional steps are discussed in the following sections.

Visual Attention: The input to the decoder consists of the visual features, represented as a set of annotation vectors corresponding to different spatial regions of the image, and the caption sequences, including the target captions. During training, each word in the caption is processed one timestep at a time.

The input to the LSTM cell includes the following and is visualized in Figure 3:

- The ground truth tokens from the target captions, y_{t-1}, multiplied by an embedding matrix E,
- The previous decoder hidden state, h_{t-1},
- The context vector, $\hat{\mathbf{z}}_t$, which is a dynamic representation of the relevant parts of the image input at time t.

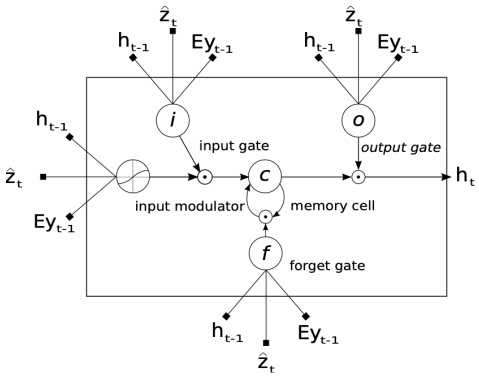

Fig. 3. An LSTM cell with inputs. [1]

Here, we deviate slightly from the original paper, where the input to the LSTM cell is $\hat{y}_{t-1}$, the previously generated word. Instead, we employ teacher forcing during training by using the ground truth word y_{t-1}. However, during inference, the LSTM cell relies on its own predictions to generate the caption and therefore the input to the LSTM cell is $\hat{y}_{t-1}$.

The context vector $\hat{\mathbf{z}}_t$ is calculated dynamically at each time step t. To determine which spatial region of the image to focus on, an energy score e_{ti} is calculated for each location i as follows:

$$e_{ti} = f_{\text{att}}(\mathbf{a}_i, \mathbf{h}_{t-1}), \tag{1}$$

where $\mathbf{a}_i$ represents the annotation vector at location i, and $\mathbf{h}_{t-1}$ is the previous hidden state of the decoder.

The attention weights α_{ti} are then computed using a softmax function to normalize the energy scores:

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{L} \exp(e_{tk})}. \tag{2}$$

These weights α_{ti} determine the relative importance of each annotation vector $\mathbf{a}_i$.

$$\hat{\mathbf{z}}_t = \phi\left(\{\mathbf{a}_i\}, \{\alpha_{ti}\}\right). \tag{3}$$

The context vector $\hat{\mathbf{z}}_t$ serves as a dynamic summary of the image features at timestep t. The specific way it is calculated depends on the type of attention mechanism employed—soft attention or hard attention. The difference lies in how the weights α_{ti} are interpreted and applied, which will be elaborated on in the following sections.

Hard Attention: For the decoder with hard attention, the model selects one single area of the image to focus on at each time step, ignoring the rest of the image to predict the next word. The attention location s_t at time step t is chosen by sampling from a multinoulli distribution over the attention probabilities α_i. The probabilities are computed for each image region $\mathbf{a}_i$, and are dependent on the image features, or

context, and the current hidden state h_i from the LSTM layer in the decoder. Then, we can define the attention probabilities and energy scores as seen in equations 2 and 1 above. We then obtain a probability distribution that sums to 1 across all regions by passing the energy score through a softmax function. The attention location denoted s_t is sampled from this distribution, known as a Multinoulli distribution, at each time step.

$$s_t \sim \text{Multinoulli}(\alpha) \tag{4}$$

At this location, the selected context vector is used to generate the next word in the caption.

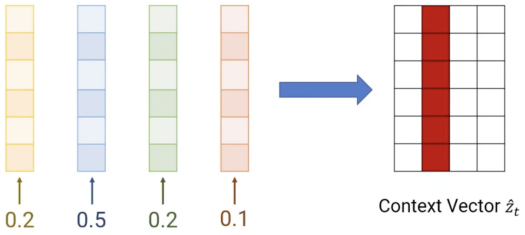

Fig. 4. Context Vectors to Hard Attention [8].

Hard Attention Training: During model training, we apply a Monte Carlo sampling method to estimate the gradient of the log-likelihood with respect to the model parameters. The attention location s_t, defined above, is a latent variable that we cannot directly observe. Hence, the stochastic and non-differentiable nature of the hard attention mechanism. To navigate this, we use the REINFORCE algorithm, which allows us to estimate the gradient of the expected log-likelihood by sampling from the attention distribution. The log probability of selecting a particular attention location $\mathbf{y}$ is

$$\log P(s_t = \mathbf{y}) = \log \alpha_{\mathbf{y}}, \quad \mathbf{y} \in \{1, 2, \ldots, L\} \tag{5}$$

which we also add a small epsilon to so we can avoid NaN or inf values.
Then, we use the REINFORCE algorithm for estimation, given by

$$\begin{aligned} L_{\text{REINFORCE}} &= -\mathbb{E}_{q(s_t)}\left[\log P(s_t|\mathbf{a})\right] \\ &= -\mathbb{E}_{q(s_t)}\left[\log \alpha_{s_t} - \log \hat{\alpha}_{s_t}^{\text{avg}}\right. \end{aligned} \tag{6}$$

where $\hat{\alpha}_{s_t}^{\text{avg}}$ is the moving average of the log probability. This moving average helps to reduce the variance of the gradient estimate, which is important since we want our estimation to be as accurate as possible. We update the moving average at each step as the paper suggests,

$$\hat{\alpha}_{s_t}^{\text{avg}} = 0.9 \cdot \hat{\alpha}_{s_t}^{\text{avg}} + 0.1 \cdot \log \alpha_{s_t} \tag{7}$$

We also include an entropy term in the loss function for regularization. The term encourages the model to explore, which helps with overfitting. The entropy is then computed as

$$Ent(\alpha) = -\sum_{i=1}^{L} \alpha_i \log \alpha_i \tag{8}$$

Finally, the total loss function in training combines the REINFORCE loss and the entropy for the total loss. Our final loss in training is then

$$L = L_{\text{REINFORCE}} + Ent(\alpha) \tag{9}$$

Hard Attention Inference: During inference, the hard attention layer adds extra steps in an effort to generate more coherent and readable captions. Importantly, we add a temperature scaling approach, which is a common control parameter in large language models "...that influences the randomness of the language model's output. It affects the probability distribution of the next word in a sequence" [9]. This approach, which was popularized in the 2021 paper "Image Captioning by Committee Consensus" [6], allows us the adjust the sharpness of the attention, so that the approach does not need to be greedy.

$$\alpha_i' = \frac{\alpha_i}{T}, \quad T = 2 \tag{10}$$

The temperature essentially scales the attention probabilities before passing them to the multinoulli sampling distribution to extract the vector that will generate the next word.

Soft Attention: Soft attention highlights the relative importance of different parts of the image. Regions with higher attention weights contribute more to the context vector, which is computed as a weighted sum of the annotation vectors. Specifically, the context vector at time t is given by:

$$\hat{\mathbf{z}}_t = \sum_{i=1}^{L} \alpha_{t,i} \mathbf{a}_i \tag{11}$$

where $\alpha_{t,i}$ represents the attention weight for the annotation vector $\mathbf{a}_i$ at time t, and L is the total number of spatial locations in the image, this is illustrated in Figure 5.

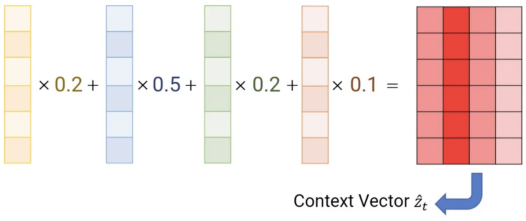

Fig. 5. Context vectors and the soft attention mechanism [8].

To achieve this, the visual features are first projected using a dense layer, while the decoder's current hidden state is projected separately. These projections are then combined through a non-linear activation function, $\tanh$, to capture their interaction. Attention scores are computed via another dense layer, and a softmax operation is applied to normalize these

scores into attention weights. These weights determine the relative importance of each spatial feature in the image. The context vector is subsequently obtained as a weighted sum of the visual features, with the attention weights serving as coefficients.

The resulting context vector, combined with the word embedding of the current word, serves as the input to the LSTM. At each timestep, the LSTM updates its hidden state and cell state based on this combined input, allowing the decoder to refine its predictions iteratively.

By employing the attention mechanism, the decoder can selectively focus on specific spatial regions of the image, leading to improved alignment between the image content and the generated words. To further enhance the generalization capability of the model, dropout regularization is applied at multiple stages, including the embedding layer, LSTM output, and final fully connected layers.

IV. Model Training

A. *Pre-processing*

We loaded captions from the `Flickr8k.token.txt` file and grouped them by image ID. To ensure one caption per image, we retained the first caption for each image, resulting in 8,092 unique image-caption pairs. Captions were cleaned by converting to lowercase, removing punctuation, trimming spaces, and adding `<START>` and `<END>` tokens (e.g., *"A child in a pink dress..."* → `<START> a child in a pink dress <END>`). Using predefined splits, captions were divided into training (6,000), validation (1,000), and test (1,000) sets, which resulted in 8,000 images and captions. A vocabulary of the most frequent words was built from training captions, including special tokens `<PAD>` and `<UNK>`, resulting in a vocabulary size of 3,869. Captions were tokenized using the vocabulary, replacing out-of-vocabulary words with `<OOV>`. For example, `<START> a child in a pink dress <END>` was tokenized as `[3, 2, 41, 5, 2, 73, 137, 4]`. These steps prepared the data for training and validation in the image captioning pipeline.

Next, we implemented a data generator to produce batches of data for training. It took the image features, tokenized captions, and a maximum caption length, and a batch size as inputs. The function shuffled the image IDs to introduce randomness and iterated through each image to create input-output pairs by progressively splitting each caption into an input sequence and a target sequence. The sequences were padded to the specified maximum length, as seen in Fig 6. This ensures that all sequences are are the same length, enabling consistent input dimensions.

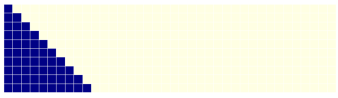

Fig. 6. Captions build sequentially for training for each image

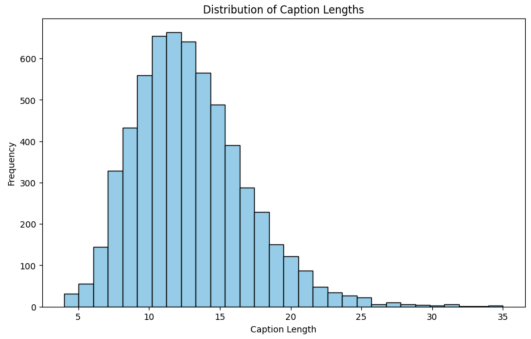

Fig. 7. Caption Distribution

In our training data, the maximum length caption is 35. However, the 75th percentile for captions is 15, as seen in Fig 7.

That means that 25% of our captions are padded with up to 20 zeros even for the full caption. Since we did not want our model to struggle with too many zeros, especially with the small training dataset, we decided to limit our captions to those with 15 words or less. Thus, any captions less than length 15 are appended with zeros, and any greater than length 15 are dropped. Once the specified batch size was reached, the generator yielded a batch containing image features, input captions, and target captions, facilitating memory-efficient model training.

B. *Training*

During training, the model learns to map input image features and partial caption sequences to their corresponding next words in the caption. The batches of image features are paired with their corresponding target outputs (i.e., the next word in the sequence). We train the models with attention for 10 epochs and with the same hyperparameters, for comparison's sake, and the generic model for 8 epochs since it starts to overfit after 8.

The training process employs teacher forcing, where the ground truth word at the current time step is fed as input to predict the next word. This approach accelerates convergence by helping the model focus on learning the mapping between inputs and outputs without compounding prediction errors during training. For each batch, the model processes the image features through a dense layer to extract higher-level information, while the caption sequences are embedded into a vector space and passed through an LSTM to capture temporal dependencies. In case there was an attention mechanism it dynamically focuses on relevant parts of the image, improving context awareness. The combined outputs of these components are used to predict the next word in the caption.

The loss, computed using sparse categorical cross-entropy, quantifies the difference between the predicted word probabilities and the ground truth. The model parameters were updated via backpropagation and the Adam optimizer to minimize the loss. Though the paper used RMSProp as the optimizer, we

Fig. 8. Generic Model Example

opted for the more state-of-the-art Adam optimizer. Validation is performed at the end of each epoch to monitor generalization, ensuring the model learns effectively without overfitting.

C. *Caption Generation*

During the caption generation process, the trained model used the visual features of the images and tokenized captions to produce human-readable descriptions. The generation incorporated a temperature parameter, which controls the randomness of the predictions. A temperature value below 1 encourages the model to generate more deterministic and conservative captions, favoring high-probability words, whereas a value above 1 increases randomness, leading to more diverse but potentially less accurate outputs. As described in the Hard Attention Inference section, we used a temperature value of 2 to balance creativity and accuracy, allowing the model to generate varied captions while maintaining contextual relevance. This method enhances the model's ability to generalize to diverse datasets while avoiding overly repetitive outputs.

V. Model Results

TABLE I
PERFORMANCE COMPARISON OF IMAGE CAPTIONING MODELS

Model	BLEU	METEOR	Validation Accuracy
Generic	0	12.9	0.3459
Soft Attention	0	3.5	0.7818
Hard Attention	0	7.7	0.8539

VI. Results Discussion

Generic vs Soft Attention vs Hard Attention: Surprisingly, we had the best captions with our generic model, see Fig 8 (no visual attention). Though the accuracy was lower, the captions more closely matched the images, and did not seem as random or have as many repeating words. Our hypothesis is that since we have a small dataset, the generic model's lower complexity matched better with the data. The hard and soft attention frameworks are most likely too complex for the small subset of data that we used.

Paper Comparison: Compared to the original paper, our model did not perform well at all. We do believe that our framework is valid, and that the performance would be better using

a. All captions from Flickr8
b. Flickr30
c. MS COCO
d. Any/all of the above with augmented data

Unfortunately for the scope and timeline of the project, and the resources available to us, we were not able to realize the full potential of our models.

With the small data, the BLEU and METEOR scores were virtually 0, which is clearly very bad compared to the impressive results from the paper, with BLEU up to 67 with Flickr8.

Challenges:

Data and Resources: Because of the memory required for image data, we opted to only use the Flickr8 data for our purposes, which has 8,000 total images and one caption per image. This allowed us to explore image captioning models efficiently, testing out different methodologies, parameters, and model frameworks. However, this data is quite small compared to many state-of-the-art models, such as the MS COCO dataset used in the original paper. Though the size of such datasets require multiple days of training with strong GPU power, which was not feasible for our project, they do add valuable information, resulting in better model results. The smaller Flickr8 data could lead to overfitting, as it is not enough data to support the complexity of our model.

Repeating Captions: Since the framework for the attention models requires data that is designed to predict the words in the captions sequentially, we have many copies of each image in our training data. Specifically, we have an input for each word in each caption for each of the images. We use the maximum length caption as 15 words, and thus we have $15 * 6000 = 90,000$ total inputs in training, while only having 6,000 training images. We believe that the model learns the training data very well, but gets confused by how the captions are built. Because of this structure, the model wants to predict the same word sequentially, since it is used to seeing the same words in training. Thus, our model often generates captions with repeating words.

VII. Conclusion

In conclusion, from this project we learned a great deal about a new type of model, and we both worked for the first time on a true text generation project. The biggest challenge was certainly working with large data, and having to use smaller data in order to run our models efficiently. We are eager to continue working on this project beyond this class, and hopefully use more data to increase our model performance.

TABLE II
WORK SPLIT

Section	Person
EDA	Both
Data preprocessing	Both
Model without Attention	Both
Soft Attention	Anouck
Hard Attention	Lucy
Report Writing	Both

In our github, we worked in branches, and then brought our code together and only pushed our final code to the main branch. You can explore our code journey in our individual branches, while the main branch contains the final, executable code.

ACKNOWLEDGMENT

We would like to thank the authors of the "Show, Attend and Tell" paper for the for their groundbreaking work, which served as a significant source of inspiration for our project. We would also like to express our appreciation to Prof. Zoran Kostic for designing and delivering an engaging and insightful course, which taught us how to complete a project like this and that we truly enjoyed.

REFERENCES

[1] K. Xu, J. Lei Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention," in *International Conference on Machine Learning (ICML)*, 2015.

[2] H. Tsaniya, C. Fatichah, and N. Suciati, "Transformer Approaches in Image Captioning: A Literature Review," in *2022 14th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2022, pp. 1–6, doi: 10.1109/ICITEE56407.2022.9954086.

[3] H. Sharma and D. Padha, "From Templates to Transformers: A Survey of Multimodal Image Captioning Decoders," in *2023 International Conference on Computer, Electronics Electrical Engineering Their Applications (IC2E3)*, 2023, pp. 1–6, doi: 10.1109/IC2E357697.2023.10262494.

[4] N. Rotstein, D. Bensaïd, S. Brody, R. Ganz, and R. Kimmel, "FuseCap: Leveraging Large Language Models for Enriched Fused Image Captions," in *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2024, pp. 5677–5688, doi: 10.1109/WACV57701.2024.00559.

[5] D. Sharma, C. Dhiman, and D. Kumar, "Evolution of visual data captioning Methods, Datasets, and evaluation Metrics: A comprehensive survey," *Expert Systems with Applications*, vol. 221, 2023, Art. no. 119773. [Online]. Available: https://doi.org/10.1016/j.eswa.2023.119773

[6] S. S. Nallapaneni and S. Konakanchi, "A Comprehensive Analysis of Real-World Image Captioning and Scene Identification," *arXiv preprint arXiv:2308.02833*, 2023. [Online]. Available: https://arxiv.org/abs/2308.02833

[7] S. Bhalerao, "VGG Net Architecture Explained," *Medium*, 2023. [Online]. Available: https://medium.com/@siddheshb008/vgg-net-architecture-explained-71179310050f

[8] Halfling Wizard, "Show, Attend And Tell - Paper Explained," *YouTube*, 2021. [Online]. Available: https://www.youtube.com/watch?v=y1S3Ri7myMg&t=509s

[9] N. Samsudin, "Temperature and Top-p Sampling in LLMs," 2024. [Online]. Available: https://medium.com/@noufalsamsudin/temperature-and-top-p-sampling-in-llms-453bcd888f13#:~:text=What%20is%20Temperature?,but%20increasing%20reliability%20and%20coherence