Image Caption Generator

Farid Davletshin [1] **Fakhriddin Tojiboev** [1] **Albert Sayapin** [1] **Dmitriy Gilyov** [1] **Lina Bashaeva** [1] **Olga Gorbunova** [1] **Evgeniy Garsiya** [1] **Hai Le** [1]

Abstract

Generation of image content description is a fundamental problem in the artificial intelligence field that connects computer vision and natural language processing. In this paper, we present a comparison of models that lie in an encoder-decoder framework. We specifically experimented with different classes of backbones like: Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Transformers. To have a justifiable and consistent experiment, we trained/validated all frameworks along with its backbone on Flickr8k dataset. Moreover, we evaluated our models on a subset of COCO to assess the performance. Our results are as follows:

- The best model is Densenet161 + Transformer and it shows BLEU@1 score 62.57.
- Experiments showed that such model configurations like CNN + LSTM are better in terms of computation time but they cannot achieve the same quality as Transformer-based architectures.

Github repo: https://github.com/tojiboyevf/image_captioning
Video presentation: https://youtu.be/pe4YZveHoqU

1. Introduction

Automatically generating captions of images is a task that reside within the heart of computer vision. This problem is fundamentally complex due to its nature. Not only must the model be powerful enough such that it is capable of recognizing the objects within an image, it must also be capable of describing and expressing the objects' relationship in a natural language. It is an important and interesting challenge for machine learning/deep learning algorithms, as it mimics humans' impressive ability of transforming visual cues into descriptive languages.

[1]Skolkovo Institute of Science and Technology, Moscow, Russia. Correspondence to: Farid Davletshin <@skoltech.ru>.

Final Projects of the Deep Learning 2022 Course, Skoltech, Moscow, Russian Federation, 2022.

Application of the automatic generation of captions of images spans across many aspects of our daily lives. Notably, this tasks coincides with several medical usage, such as medical diagnosis, virtual assistants for visually impaired individuals/disabled individuals, etc... Other usage includes image indexing, improvement of search engines, recommendation system for editing software, and many more.

The practicality of image caption as a topic also comes with its obvious challenges. Despite that, there has been a continual surge of research interest in attacking this blend of computer vision and natural language processing. Recent advances in neural network studies resulted in performance increase in using the classic encoder-decoder framework. Commonly, this combination includes convolutional neural networks (convnets) to obtain vectorial representation of images and recurrent neural networks to decode those representations into natural language sentences. In simpler terms, the encoder encodes the input image into an intermediate representation of the information in the image. Then, the decoder decodes the encoder's output into a descriptive text sequence.

In this paper, we discuss and experiment with widely used encoder and decoder framework. Several combinations of backbones were used, which includes: Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Transformers. To keep our experiment and comparison consistent, we trained/validated all of our models and its variations on the Flickr8k. There were initial contemplation on relying on COCO as the dataset of choice. However, due to our limitations in computational resources, that was deemed unfeasible. Despite that, we still used a subset of COCO to further evaluate the performance of our models.

2. Related work

In this section, we provide the relevant background on previous work on image caption generation and attention. Recently, several methods have been proposed for generating image descriptions. Many of these methods are based on recurrent neural networks and inspired by the success of sequence to sequence training with neural networks for machine translation (Bahdanau et al., 2015), (Cho et al., 2014). One major reason image caption generation is well suited

to the encoder-decoder framework (Cho et al., 2014) of machine translation is because it is analogous to "translating" an image to a sentence.

The first attempt to use neural networks for the generation of captions was (Kiros et al., 2014a), who proposed a multimodal log-bilinear model that was biased by features from the image. This work was later followed by (Kiros et al., 2014b) whose method was designed to explicitly allow a natural way of doing both ranking and generation. (Mao et al., 2014) took a similar approach to generation but replaced a feed-forward neural language model with a recurrent one. Both (Vinyals et al., 2014) and (Donahue et al., 2014) use LSTM RNNs (Hochreiter & Schmidhuber, 1997) for their models. Unlike (Kiros et al., 2014a) and (Mao et al., 2014) whose models see the image at each time step of the output word sequence, (Vinyals et al., 2014) only show the image to the RNN at the beginning.

All of these works represent images as a single feature vector from the top layer of a pre-trained convolutional network. (Karpathy & Fei-Fei, 2014) instead proposed to learn a joint embedding space for ranking and generation whose model learns to score sentence and image similarity as a function of R-CNN object detections with outputs of a bidirectional RNN. (Fang et al., 2014) proposed a three-step pipeline for generation by incorporating object detections. Their model first learn detectors for several visual concepts based on a multi-instance learning framework.

Speaking of more recent models we should point out (Mokady et al., 2021) where the authors use CLIP encoding from (Radford et al., 2021) as a prefix to the caption, by em- ploying a simple mapping network, and then fine-tunes a language model to generate the image captions. Their key idea is that together with a pre-trained language model GPT2 (Radford et al., 2019), they obtain a wide understanding of both visual and textual data. The current SOTA is OFA (Wang et al., 2022), where the authors pursue a unified paradigm for multimodal pretraining. It is a unified multimodal pretrained model that unifies modalities (i.e., cross-modality, vision, language) and tasks (e.g., image generation, visual grounding, image captioning, image classification, text generation, etc.) to a simple sequence-to-sequence learning framework based on the encoder-decoder architecture. Experimental results show that OFA achieves new state-of-the-arts on image captioning task.

3. Models and algorithms

In this section, we discussed the extensive architecture details of the different encoder-decoder frameworks and backbone variations that were used. Mainly, we experimented with two types of encoders (CNN and Visual Transformer) and two type of decoders (RNN - LSTM based and Transformer). The main sections include:

- CNN and RNN (LSTM based)
- Visual Transformer and RNN (LSTM based)
- CNN and Transformer
- Visual Transformer and Transformer

3.1. Encoder-Decoder architecture using CNN coupled with RNN

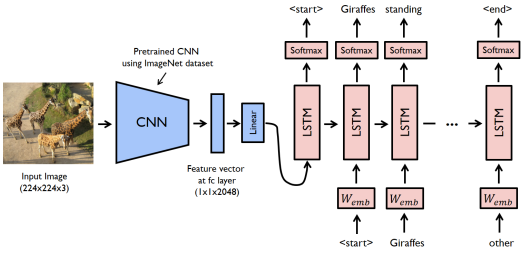

Figure 1. Model based on CNN encoder and LSTM decoder.

From (Vinyals et al., 2014) the main objective of the image caption generator is to maximize the probability of the correct description given the image by using the following formulation:

$$\theta^* = \arg\max_x \sum_{(I,S)} \log(p(S|I;\theta)) \tag{1}$$

where θ are the model weights, I is an image, S is a correct image description.

As S represents any sentence, its length is unbounded. Thus, it is common to apply the chain rule to model the joint probability over $S_0, \ldots, S_N$, where N is the length of this particular example as:

$$\log(p(S|I;\theta)) = \sum_{i=0}^{N} \log(p(S_i|I, S_0, \ldots, S_{i-1};\theta)) \tag{2}$$

During training, (I, S) is a training example pair, and we optimize (1) over the whole training set using some optimizer.

Similar to the authors, we used a Convolutional Neural Network for the representation of images. In other words, the CNN encoder is used for feature extraction of the images. Thus, it allows for rich feature representations from the image using convolutional and pooling layers paired with batch normalization and relu as a non-linear function. The words are represented with an embedding model.

For the model $p(S_i|I, S_0, \ldots, S_{i-1}; \theta)$, we use Recurrent Neural Network, where the variable number of words we condition upon up to $i-1$ is expressed by a fixed length hidden state h_i. This hidden state is updated after seeing a new input x_i by using a non-linear function f:

$$h_{i+1} = f(h_i, x_i) \tag{3}$$

where f - LSTM or GRU models, x_i - CNN image representation or word embedding.

The full model can be represented as on Figure 1:

$$x_{-1} = \text{CNN}(I) \tag{4}$$
$$x_i = W_e S_i, i = 0, \ldots, N-1 \tag{5}$$
$$p_{i+1} = \text{LSTM}(x_i), i = 0, \ldots, N-1 \tag{6}$$

where S_i - one-hot vector with dimension equal to the size of the dictionary, S_0 and S_N - special start and stop words, W_e - word embeddings matrix.

Target loss is the sum of the negative log likelihood of the correct word at each step:

$$L(I, S) = -\sum_{i=1}^{N} log(p_i(S_i)) \tag{7}$$

In inference time, we used BeamSearch to generate a sentence given an image. It iteratively considers the set of k best sentences up to time i as candidates to generate sentences of size $i+1$. Moreover, it only keeps the best k set.

For our CNN encoder, we used a pretrained model like DieT and DenseNet161. For RNN decoder, we considered both LSTM and GRU. At the end, we decided to used LSTM as as GRU did not offer any considerable improvements.

3.2. Visual transformer coupled with RNN

As opposed to a CNN encoder, the second model that we experimented with utilized a different encoder. We implemented a visual transformers encoder to extract the input images' features. Specifically, we use a pretrained Imagenet Data-efficient image Transformer (DeiT) model (Touvron et al., 2020).

For our experiment, we considered both DeiT and ViT (Dosovitskiy et al., 2020). Both of these vision transformers possesses similar architecture. However, DeiT have an additional distillation token for input token. Comparing these two architecture performances, ViT does not perform well when trained on smaller amounts of data. Thus, we decided to implement DeiT as our vision encoder.

DeiT have an additional distillation token. The architecture of this token is shown in the figure below (Figure 2).

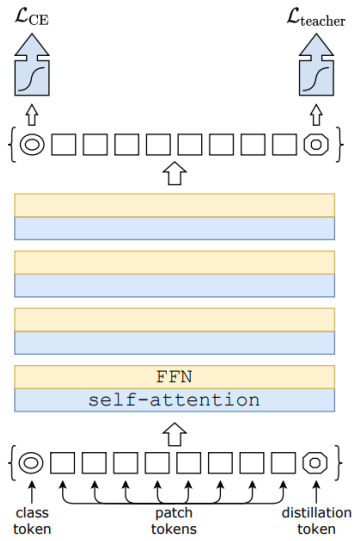

Figure 2. Distillation procedure

The distillation token interacts with the class and patch tokens through the self-attention layers. Moreover, the token is employed in a similar fashion as the class token, except that on output of the network its objective is to reproduce the (hard) label predicted by the teacher, instead of true label. Both the class and distillation tokens input to the transformers are learned by back-propagation. The distillation embedding allows the model to learn from the output of the teacher, as in a regular distillation.

The core of the encoder-decoder architecture remains the same from the first model described in the previous section

3.3. CNN coupled with Transformer

As opposed to the first two models, this model utilizes a transformer based decoder (Vaswani et al., 2017) instead of LSTM.

The introduction of a transformer based decoder eliminate the need for recursion. The focal point of transformer include mechanism like multi-head attention and positional embeddings. These features allows for the ability to process sentences as a whole and learn relationships between words. Thus, they became state-of-the-art in the sequence-to-sequence tasks. Therefore, we hypothesized model with transformers decoder will likely get better quality on image caption generation task.

3.4. Visual transformer coupled with Transformer

Lastly, this model differs from the rest of the previous models mentioned above. We utilizes Transformer for both encoders and decoders. Similar to the model above, we implemented DieT as encoder. The decoder is a transformer

based model (Vaswani et al., 2017). The architecture of the transformer based decoder is shown in the figure below.

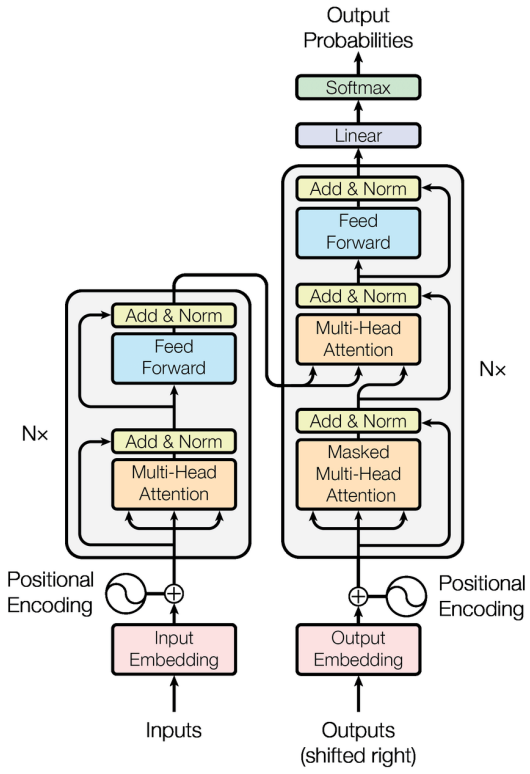

Figure 3. The Transformer - model architecture.

During transfer learning, we saved DieT's weights added one Linear layer, GLEU activation, Batch Normalization and Dropout. This structure worked noticeably better in comparison to emitting the head layer of pretrained DieT. The architecture of the Transformer decoder were kept exactly the same. However, we did change some notable hyperparameters such as number of heads and number of layers to achieve the highest score. Moreover, we tried different positional encoder for decoder. At the end, the suggested encoder from the authors performed better.

4. Experimental settings

In this section, we provided a comprehensive overview of our experiment methodology and settings.

4.1. Data

Throughout our experiments, we decided to train our models on the Flickr8k Dataset. The decision was partly due to our lack of computational power and time. Our primary goal is to implement a variety of different encoders and decoders. The objective was to compare and contrast the performance of these models. Thus, training on a smaller

Moreover, we further evaluated on models on a subset of the COCO 2014 dataset. Initially, we wanted to evaluate on the whole of COCO. Moreover, the authors also validated their model on the entire dataset. That proved infeasible due to the heaviness of this dataset.

At first we wanted to use a COCO dataset for training, so that the comparison of our and papers' results would be more clear and fair, but COCO is too heavy and takes long time to train on our computers. As our primary purpose was to implement as many models as we can. Thus, we decided to save it for these purposes and switched to Flickr8k.

Flickr8k Dataset is a new benchmark collection for sentence-based image description and search, consisting of 8,000 images that are each paired with five different captions which provide clear descriptions of the salient entities and events. The images were chosen from six different Flickr groups, and tend not to contain any well-known people or locations, but were manually selected to depict a variety of scenes and situations

4.2. Preprocessing

For training images, we used a composition of the following transformations:

- Resize (the resizing size depended on the different backbones used as each require different input dimensions; sizes between is was between 256 and 299)
- Random crop
- Random horizontal flip
- Normalization

4.3. Parameters and configurations

We used pre-trained GloVe (Pennington et al., 2014) with sizes 50, 100, 200, 300 as embedding in order to represent words in embedding space. We tried different embedding sizes and we chose embedding size depending on model. Each embedding size showed different results for each model.

4.4. Metrics

For measuring quality of our models we choose a BLEU Score, or the Bilingual Evaluation Understudy. It is a score for comparing a candidate translation of text to one or more reference translations. The BLEU score was proposed by Kishore Papineni, et al. in their 2002 paper (Papineni et al., 2002).

The approach works by counting matching n-grams in the candidate translation to n-grams in the reference text, where

1-gram or unigram would be each token and a bigram comparison would be each word pair. The comparison is made regardless of word order. The more the matches, the better the candidate translation is.

BLEU's output is always a number between 0 and 1.

We set n equal to 4, meaning that we counted only 1-grams and 2-grams. The reason of choosing such n is that the captions are not very big, especially they contain only one sentence, but not a paragraph or even a text. The second reason is that we want our captions to be reasonable, but not exactly matched to the primer description, that's why it's not very reasonable to count for longer grams.

We evaluated the BLEU scores of single generated caption with respect to 5 references for both datasets Flickr and COCO. Choosing 5 references is mostly used in many BLEU evaluation that's why we also chose 5 references.

5. Results

In this section, we will discuss the results gathered from training our models on Flickr8k and evaluating on COCO val2014. The performance of models are smaller in COCO val2014, but we think it's due to that we didn't train models on big dataset COCO train2014.

5.1. Qualitative Results

Table 1. BLEU scores of different models on COCO val2014 dataset, %

Model	BLEU1	BLEU2	BLEU3	BLEU4
DenseNet161+LSTM(69ep)	49.33	23.25	11.70	9.46
VGG16+LSTM(3ep)	46.71	23.75	12.25	8.39
VGG16+Transformer	25.55	10.04	4.1	2.04
DenseNet161+Transformer	**55.38**	**30.71**	**17.09**	9.79
DeiT+LSTM	45.73	22.04	11.14	9.12
DeiT+Transformer	53.09	29.76	16.92	**9.95**
InceptionV3+Transformer	49.14	26.49	14.21	8.11

Table 2. BLEU scores of different models on Flickr8k test set, %

Model	BLEU1	BLEU2	BLEU3	BLEU4
DenseNet161+LSTM(69ep)	55.27	30.76	17.11	10.23
VGG16+LSTM(3ep)	55.41	34.34	21.13	13.29
VGG16+Transformer	52.76	33.04	20.27	12.39
DenseNet161+Transformer	**65.98**	**44.79**	30.04	19.75
DeiT+LSTM	53.48	31.06	17.61	10.61
DeiT+Transformer	62.57	44.09	**35.11**	**29.80**
InceptionV3+Transformer	60.19	39.19	25.70	16.70

Analysis of Qualitative Results:

Overall, the best performing model is Densenet161 when paired with Transformer. Just like we hypothesized, the Transformer decoder increased the accuracy of the model. This is due to several features that comes with transformer; mainly multi-headed attention and positional encodings.These features allows for the ability to process sentences as a whole and learn relationships between words.

5.2. Generated captions

5.2.1. InceptionV3 and Transformer

Figure 4. Good image caption

Figure 5. Bad image caption

Analysis of Captions by InceptionV3 + Transformer:
TODO

5.2.2. DENSENET161 AND TRANSFORMER

Figure 6. Good image caption

Figure 7. Bad image caption

Analysis of Captions by Densenet161 + Transformer:

In the good example, the predicted caption is a man riding a bike on the street. This is quite good. The model did fail to recognize that there are multiple men riding the bikes/motorcycles. Despite that, it's still a good prediction.

5.2.3. DEiT AND TRANSFORMER

Figure 8. Good image caption

Figure 9. Bad image caption

Analysis of Captions by DEiT + Transformer:

The performance for DEiT paired with Transformer is satisfactory. The the best example produced a BLEU@4 score of 1.0 while the worst example produced 0.08. In other generated captions, this model generates captions in average with BLEU@1=0.6257 and BLEU@4=0.2980 on Flickr. Sometimes model can't detect the gender of people, detects wrongly background or animals . But in general most of the captions are meaningful.

5.2.4. DENSENET161 AND LSTM

Figure 10. Good image caption

Figure 11. Bad image caption

Analysis of Captions by Densenet161 + LSTM:

From the good example, we can see that the model achieved pretty high BLEU@4 score 0.39. That is why we expect similar text description as a result. The model was able to identify "a dog running through the grass", but at the same time it took a man incorrectly for another white dog. As for the bad example, there is no surprise as BLEU@4 score 0.06 and the model captured "red shirt", "red ball" incorrectly and the overall sense of the sentence was inaccurate.

5.2.5. VGG16 AND LSTM

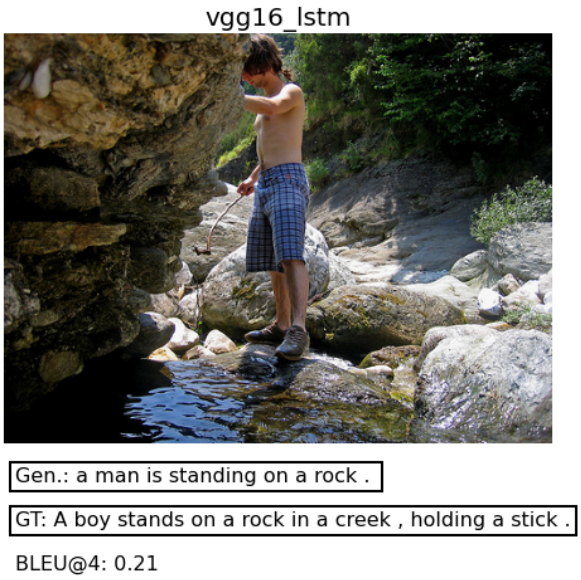

Figure 12. Good image caption

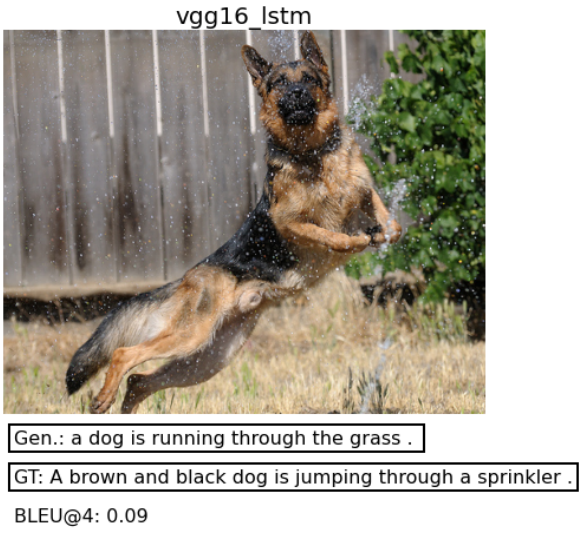

Figure 13. Bad image caption

Analysis of Captions by VGG16 + LSTM:

Despite the fact that the model achieved BLEU@4 score 0.21, it's still less than the Densenet161 + LSTM's BLEU@4 score. This is due to the improvement of textual description. It grasped the main idea that "a man is standing on a rock". On the other hand, BLEU@4 score for the bad example was 0.09. Overall model captured the main idea using the other words; thus resulting in mediocre scores.

5.2.6. DEiT AND LSTM

Figure 14. Good image caption

Figure 15. Bad image caption

Analysis of Captions by DEiT + LSTM:

The performance of DEiT when paired with LSTM showed good BLEU scores for both the good and bad examples. For the good example, the BLEU scores is comparable to Densenet161 + LSTM. However, the bad example scored quite high. Even though the model predicted the completely wrong situation, it is understandable why it intepreted the input picture as a dog (black skier) digging a hole.

6. Conclusion

In this project, we explored several implementation approaches to the problem of automatic generation of captions for images. Specifically, we successfully implemented and experimented with several backbones based on the classical encoder-decoder framework. We tested two different encoders - Convolutional Neural Networks and Visual transformers. We further tested two different encoders - Residual Neural Networks and Transformer based. For both CNN and RNN, we utilized pretrained models such as Inceptionv3, VGG16, and Densenet161 for CNN; LSTM for RNN.

After evaluating our models with COCO val2014, we found that the best performing model was Densenet161 paired with Transformer. This was our hypothesis as Transformer possessed features that are deemed performance improving across recent literature/research. The introduction of a transformer based decoder eliminate the need for recursion. The focal point of transformer include mechanism like multi-head attention and positional embeddings. These features allows for the ability to process sentences as a whole and learn relationships between. Thus, they became state-of-the-art in the sequence-to-sequence tasks. Sensibly, it performed better than the LSTM decoders. However, the LSTM decoders did provide a quicker computation time, albeit at the cost of performance.

References

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2015.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014. URL http://arxiv.org/abs/1406.1078.

Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. *CoRR*, abs/1411.4389, 2014. URL http://arxiv.org/abs/1411.4389.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. URL https://arxiv.org/abs/2010.11929.

Fang, H., Gupta, S., Iandola, F. N., Srivastava, R. K., Deng, L., Dollár, P., Gao, J., He, X., Mitchell, M., Platt, J. C., Zitnick, C. L., and Zweig, G. From captions to visual concepts and back. *CoRR*, abs/1411.4952, 2014. URL http://arxiv.org/abs/1411.4952.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Karpathy, A. and Fei-Fei, L. Deep visual-semantic alignments for generating image descriptions. *CoRR*, abs/1412.2306, 2014. URL http://arxiv.org/abs/1412.2306.

Kiros, R., Salakhutdinov, R., and Zemel, R. Multimodal neural language models. In Xing, E. P. and Jebara, T. (eds.), *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 595–603, Bejing, China, 22–24 Jun 2014a. PMLR. URL https://proceedings.mlr.press/v32/kiros14.html.

Kiros, R., Salakhutdinov, R., and Zemel, R. S. Unifying visual-semantic embeddings with multimodal neural language models. *CoRR*, abs/1411.2539, 2014b. URL http://arxiv.org/abs/1411.2539.

Mao, J., Xu, W., Yang, Y., Wang, J., Huang, Z., and Yuille, A. Deep captioning with multimodal recurrent neural networks (m-rnn). 2014. doi: 10.48550/ARXIV.1412.6632. URL https://arxiv.org/abs/1412.6632.

Mokady, R., Hertz, A., and Bermano, A. H. Clipcap: CLIP prefix for image captioning. *CoRR*, abs/2111.09734, 2021. URL https://arxiv.org/abs/2111.09734.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040.

Pennington, J., Socher, R., and Manning, C. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable

visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021. URL https://arxiv.org/abs/2103.00020.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. *CoRR*, abs/2012.12877, 2020. URL https://arxiv.org/abs/2012.12877.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. Show and tell: A neural image caption generator. *CoRR*, abs/1411.4555, 2014. URL http://arxiv.org/abs/1411.4555.

Wang, P., Yang, A., Men, R., Lin, J., Bai, S., Li, Z., Ma, J., Zhou, C., Zhou, J., and Yang, H. Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. *CoRR*, abs/2202.03052, 2022. URL https://arxiv.org/abs/2202.03052.

A. Contribution Section

Explicitly stated contributions of each team member to the final project.

Farid Davletshin

- Worked on Transformer architecture implementation;
- Worked on VGG16 + Transformer architecture implementation
- Worked on Inception_v3 + Transformer architecture implementation
- Literature review
- Code review

Fakhriddin Tojiboev

- Prepared diet_transformer.ipynb notebook;
- Worked on implementation of DieT and Transformer;
- Conducted experiments with diet+transformer;
- Prepared dataloader for COCO dataset;
- Wrote some parts of section 4 and section 3;
- Literature review;
- Code review;

Albert Sayapin

- Prepared **cnn_lstm.ipynb** notebook with full pipeline for evaluation and captions generation;
- Worked on implementation of DenseNet161, VGG16 and LSTM;
- Conducted experiments with Densenet161+LSTM and VGG16+LSTM;
- Arranged "Introduction", "Related work", "Models and algorithms", "References" sections of the report;
- Set environment of the repository: packages install, data install;

Dmitriy Gilyov

- Did a research on different approaches to the image captioning task;
- Worked on implementation of LSTM + InceptionV3;
- Conducted experiments with LSTM architecture based on InceptionV3 and DenseNet backbones;
- Co-wrote and edited the report

Lina Bashaeva

- Worked on implementation on BLEU metrics and InceptionV3+LSTM
- Worked on transformer-based model, was searching on its architecture and different implementations
- Conducted experiments with InceptionV3+LSTM
- Prepared some parts of report, presentation and video recording

Olga Gorbunova

- Worked on implementation of DenseNet161 and Transformer;
- Conducted experiments with Densenet161 and transformer;
- Literature review;
- Code review;

Evgeniy Garsiya

- Worked on DieT + LSTM network implementation
- Conducted experiments with DieT+LSTM
- Searching for information for slides and speech.
- Prepared several parts of presentation.
- Video recording

Hai Le

- Worked the implementation of Inceptionv3 and LSTM
- Conducted experiments with Inceptionv3/Vit and LSTM and code review
- Co-Write the Introduction, Models, Results, and Conclusion.
- Prepared some parts of the presentation.

B. 3rd party code list

- https://github.com/sauravraghuvanshi/Udacity-Computer-Vision-Nanodegree-Program/tree/master/project_2_image_captioning_project
- https://github.com/siddsrivastava/Image-captioning
- https://github.com/iMeleon/EECS-498-007-598-005-solutions/tree/master/2020/A4_solution
- https://github.com/ruotianluo/ImageCaptioning.pytorch
- https://github.com/Subangkar/Image-Captioning-Attention-PyTorch
- https://pytorch.org/tutorials/beginner/transformer_tutorial.html