Image to LaTeX

Matvey Skripkin [1] **Nikita Ligostaev** [1] **Dmitrii Baluev** [1] **Nikolay Kalmykov** [1] **Antonina Kurdyukova** [1]

Abstract

LaTeX is a software system for document preparation. It is widely used in academia for the communication and publication of scientific documents in many fields. But preparation materials using LaTeX requires from the writer a lot of time and effort. So, the problem of image-to-markup generation is solved in this work. Images with formula are translated to LaTeX code. The proposed model is based on the transformer architecture. The method for comparison is based on common CNN model.

Github repository: link to the project Github repository.

1. Introduction

LaTeXis one of the most successful and amazing free software projects ever done. With LaTeX you can have a really high quality typesetting document. It is highly recommended to use LaTeX in the following cases:

- scientific articles and other manuscripts;
- when you need to type formulas;
- if you work with abundant bibliography;
- if you need outprints with figures using the best quality possible;
- several output formats for one entry point;
- etc.;

But using LaTeX could be quite challenging for the writer and require a lot of time and effort.

[1]Skolkovo Institute of Science and Technology, Moscow, Russia. Correspondence to: Matvey Skripkin <Matvey.Skripkin@skoltech.ru>, Nikita Ligostaev <Nikita.Ligostaev@skoltech.ru>, Dmitrii Baluev <Dmitrii.Baluev@skoltech.ru>, Nikolay Kalmykov <Nikolay.Kalmykov@skoltech.ru>.

Final Projects of the Machine Learning 2023 Course, Skoltech, Moscow, Russian Federation, 2023.

So, the problem of converting images of mathematical formulas to LaTeX arises. It combines challenges both from computer vision and natural language processing, close to the recent breakthroughs in image captioning. Our system would take as input a formula (the image) and would generate a list of LaTeX tokens (the caption).

Little attention has been given to reconstructing LaTeX code from mathematical formulas. OpenAi listed this problem in their blog as a request for research (https://openai.com/requests-for-research). There are already exists services for image to LaTeX translation. But it is paid and has limitations on the amount of input images.

In this project we propose our own model. Our aim to improve the results compared to the existing models.

Our main contributions are:

- a fully end-to-end LATEX rendering system that could also be applied to image captioning, and more globally to any image-to-text problem;
- improving the results of existing models;
- a comparison with CNN-based model as well as an analysis of the performance of an imagecaptioning model on this particular problem.

Note that due to the fact that different formulas can produce similar or same images add a complexity to the task, compared to standard image captioning. It also raises issues about the evaluation of such models.

2. Image to LaTeX problem

The Image to LaTeX problem is a request for research proposed by OpenAI. The challenge is to build a Neural Markup Generation model that can be trained end-to-end to generate the LaTeX markup of a math formula given its image. Data for this problem consist of grayscale images as the input samples and the original markup as the target sequence. Each training/test sample (Figure 1) is comprised of an input image $\mathbf{x}$ and a corresponding target LaTeX -sequence $\mathbf{y}$ of length τ. Each word $\mathbf{y}$ of the target sequence, belongs to the vocabulary of the dataset. Denoting image dimensions as H, W and the vocabulary as a set V of K words.

$$S_0 = \sum_l \frac{1}{2\Delta_l^2} \mathrm{Tr}\, \phi_l^a \phi_{-l}^a + \sum_l \frac{1}{2\epsilon_l^2} \mathrm{Tr}\, f_l^a f_{-l}^a + \sum_r \frac{1}{g_r} \mathrm{Tr}\, \bar{\psi}_r^a \psi_r^a \, .$$

```
S _ { 0 } = \sum _ { l } \frac { 1 } { 2 \Delta _ { l } ^ { 2 } } \mathrm { T r } \, \phi _ { l } ^ { a } \phi _ { - l } ^ { a } + \sum _ { l }
\frac { 1 } { 2 \epsilon _ { l } ^ { 2 } } \mathrm { T r } \, f _ { l } ^ { a } f _ { - l } ^ { a } + \sum _ { r } \frac { 1 } { g _ { r } } \
mathrm { T r } \, \bar { \psi } _ { r } ^ { a } \psi _ { r } ^ { a } \, .

S _ { 0 } = \sum _ { l } { \frac { 1 } { 2 \Delta _ { l } ^ { 2 } } } \mathrm { T r } \, \phi _ { l } ^ { a } \phi _ { - l } ^ { a } + \sum _ { l }
 { \frac { 1 } { 2 \epsilon _ { l } ^ { 2 } } } \mathrm { T r } \, f _ { l } ^ { a } f _ { - i } ^ { a } + \sum _ { r } { \frac { 1 } { g _ { r } }
 } \mathrm { T r } \, \psi _ { r } ^ { a } \psi _ { r } ^ { a } \, .
```

Figure 1. A training sample.

Then, we represent:

$$\mathbf{x} \in \mathbb{R}^{H \times W};$$

$$V := \{\text{\LaTeX - tokens}\}, \; |V| = K;$$

$$\mathbf{y} = (y_1, \ldots, y_T), \; y_t \in 1, \ldots, K.$$

The task is to generate markup that a LaTeX compiler will render back to the original image. Therefore, our model needs to generate syntactically and semantically correct markup, by simply 'looking' at the image.

3. Related work

In the past few years, lots of breakthroughs have taken place in Computer Vision, originated by the performance of system to recognize digits (LeCun et al., 1998). Optical Character Recognition has since gained interest, with highly-accurate systems like the one of (Cireşan et al., 2010). On the task of recognizing mathematical expressions that introduces new challenges of grammar understanding, some attempts have managed to build systems by exploiting character segmentation and grammar reconstruction, like (Miller & Viola, 1998), or other syntactic-based approaches like (Chan & Yeung, 2000) and (Belaid & Haton, 1984). Techniques combining neural network and standard NLP approaches like Conditional Random Field have also proven to be very effective to recognize words in images like in (Jaderberg et al., 2014), or using convolutional neural networks like in (Wang et al., 2012).

In Natural Language Processing, sequence-tosequence models based on recurrent neural networks have forged ahead in the race to machine translation (Sutskever et al., 2014), improving by a lot the quality of translation from one language to another. The introduction of attention by (Bahdanau et al., 2014; Luong et al., 2015) eventually established a new standard in Machine Translation systems, allowing impressive performance like zero-shot translation like in (Niehues & Cho, 2017).

Advances have also been made in Image Captioning, using recurrent neural network for the most part, like in (Karpathy & Fei-Fei, 2014; Karpathy et al., 2015) 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.

Combining sequence-to-sequence with Image Captioning techniques,the authors of (Xu et al., 2015) encode the image in a fixed size vector with a CNN, and then decoding the vector step by step, generating at each step a new word of the caption and feeding it as an input to the next step. In their work, they also added an attention mechanism, enabling the decoder at each time step to look and attend at the encoded image, and compute a representation of this image with respect to the current state of the decoder (Figure 2).

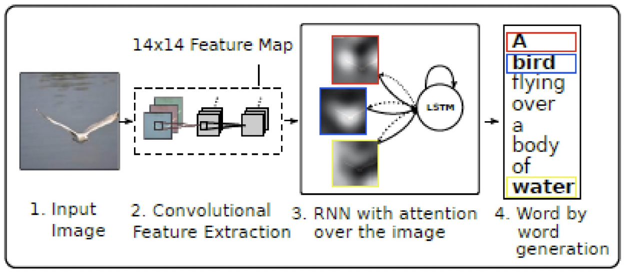

14x14 Feature Map
LSTM
A
bird
flying
over
a
body
of
water
1. Input Image
2. Convolutional Feature Extraction
3. RNN with attention over the image
4. Word by word generation

Figure 2. Attention based model for image captioning, figure from (Xu et al., 2015).

Recent work from (Deng et al., 2016) took this same approach and successfully applied it to the generation of LaTeX code from images of formulas. They used the same mechanism of soft attention as in (Luong et al., 2015), and were able to achieve as much as 77% of exact match score on the test dataset.

4. Model Description

4.1. Motivation

Our task requires both Natural Language Processing as well as Computer Vision. Then, we decided to use the Vision Encoder Decoder Model. It initializes an image-to-text

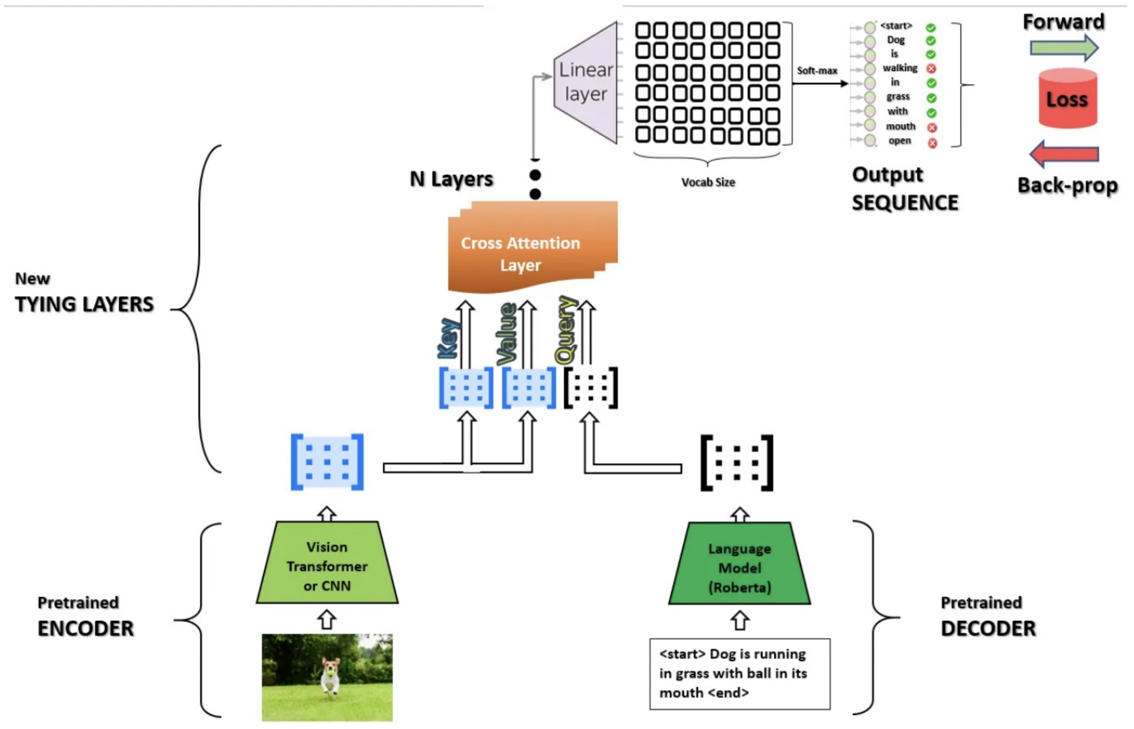


Figure 3. Vision Encoder Decoder Architecture. Reference

model with any pre-trained Transformer-based vision model as the encoder (e.g. ViT, BEiT, DeiT, Swin) and any pre-trained language model as the decoder (e.g. RoBERTa, GPT2, BERT, DistilBERT). The encoder model is used to encode the image, after which an autoregressive language model i.e. the decoder model generates the formula markup.

4.2. Model architecture

Let's describe how the Vision Encoder Decoder module connects the two models (Image Encoder and Text Sequence generator).

When the vision encoder decoder is initialized with pre-trained models, it creates an image encoder & language decoder instance and ties their embeddings together using a cross-attention layer. The encoder embeddings are used as KEY & VALUE and the decoder embeddings are used as QUERY in the cross-attention head.

During training, both image and desired formula markup are passed as inputs to the model. Using these inputs, the model is forced to generate same markup, this leads to understanding the correlation between words in captions and objects in the input images.

Final output from the linear layer is a vector of size Length of sequence $\times$ Vocabulary size (3). The elements of vector matrix (after SoftMax) are probabilities (green boxes below signifies highest probability) of the word occurring at a given position in the sequence.

At every training step, the model predicts a markup sequence which is compared with the actual sequence and the loss is propagated back to learn.

5. Experiments and Results

5.1. Dataset description

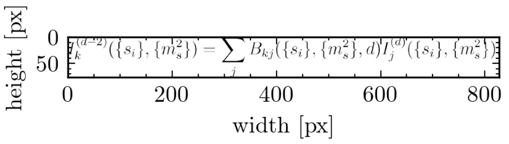
We use the following publicly available I2L-140K dataset (Singh, 2018a) for image to LaTeX problem. According to original paper, I2L-140K dataset was created by extracting single-line LaTeX math formulas from scientific papers, followed by a series of processing steps including normalization, rendering, removal of duplicates, and filtering based on word frequency, formula length, and image size. The resulting dataset was used for training, validation, and testing, and was augmented with additional samples from KDD Cup 2003 to address the issue of limited data size (Singh, 2018b).

The dataset is publicly available here (Singh, 2018a).

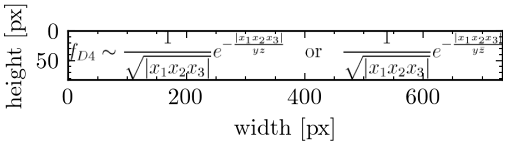
I2L-140K dataset is a collection of images of LaTeX formulas and LaTeX formulas themselves. Some samples from dataset are presented on Figure 4. Dataset contains a total of $154,944$ LaTeX images and formulas. LaTeX formulas

LaTeX formula | Image

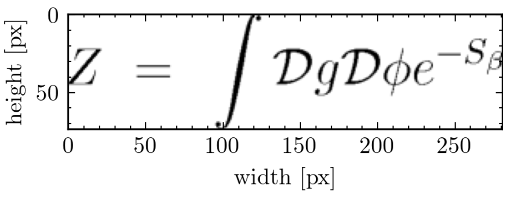
```
I _ { k } ^ { ( d -2 ) } ( \\{ s _ {
i } \\} ,\\{ m _ { s } ^ { 2 } \\} )
= \\sum _ { j } B _ { k j } ( \\{ s
_ { i } \\} ,\\{ m _ { s } ^ { 2 }
\\} ,d ) I _ { j } ^ { ( d ) } ( \\{
s _ { i } \\} ,\\{ m _ { s } ^ { 2 }
\\} ) .
```



```
f _ { D 4 } \\sim \\frac { 1 } {
\\sqrt { | x _ { 1 } x _ { 2 } x _
{ 3 } | } } e ^ { -\\frac { | x _ {
1 } x _ { 2 } x _ { 3 } | } { y z
} } \\quad \\mathrm { o r } \\quad
\\frac { 1 } { \\sqrt { | x _ { 1
} x _ { 2 } x _ { 3 } | } } e ^ {
-\\frac { | x _ { 1 } x _ { 2 } x _ {
3 } | } { y \\bar { z } } }
```



```
Z ~ = ~ \\int { \\cal D } g { \\cal
D } \\phi e ^ { -S _ { \\beta } }
```



```
b _ { j } | \\Psi _ { j } ( t )
\\rangle = \\Phi _ { j } ( t ) |
\\Psi _ { j } ( t ) \\rangle
```

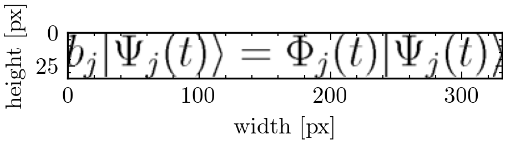


Figure 4. Sample of 4 LaTeXformulas and corresponding images that are presented in I2L-140K dataset.

length is varied from 1 to 2177 symbols.

Each image in the dataset is a 1-channel image with a resolution that depends on the length of the formula and varies from 3×3 pixels to 1020×644 pixels.

5.2. Data processing

The data processing phase involved several key steps. Firstly, we implemented a duplicates elimination technique to ensure the integrity and uniqueness of the dataset. Additionally, we enhanced the dataset by incorporating supplementary information pertaining to each image's associated formula, such as its width, height, and length.

Subsequently, we proceeded to construct a custom dataset tailored specifically for our deep learning model. This custom dataset was designed to provide both the image data and the corresponding image filenames, facilitating seamless integration and traceability during the training process.

Furthermore, we employed various augmentation strategies to enhance the diversity and robustness of our image data. These augmentations were carefully selected and applied to the images, ensuring that the model would be exposed to a wider range of realistic variations and scenarios.

5.3. Training description and evaluation

In this experiment, we utilized two encoder architecture variants: ViT Encoder and CNN Encoder.

In ViT Encoder approach the image was divided into patches of size 16×16 pixels. After passing through the encoder, we obtained the output embeddings of shape $(N, (H/patch\_size)^2, embed\_size)$. The second encoder used was CNN Encoder (ResNet101). We employed a pretrained model and only modified the first convolutional layer to accommodate the single-channel input image. We also removed the dense layers at the end and added a final convolutional layer to adjust the embedding dimension.

During training procedure the following optimizers were used: Adam and Adamax. We tested different batch sizes: $16, 32$, and 64. The learning rate used was $1e-4$.

Next metrics were used: BLEU, ROUGE-1 Precision, ROUGE-1 F-Measure, ROUGE-1 Recall, ROUGE-2 Precision, ROUGE-2 F-Measure, ROUGE-2 Recall, ROUGE-L Precision, ROUGE-L F-Measure, ROUGE-L Recall.

5.4. Results

During experiments subset of $50,500$ LaTeX formulas with ViT decoder architecture was used. However, overfitting took place (Figures 5, 6, Table 1) as expected.

In the next iteration CNN decoder on the full dataset was implemented, still statistically significant result was not achieved (Figures 5, 6, Table 1).

Finally, ViT decoder gave the best performance, it was used in testing procedure. Results are provided in Table 2 and Figure 7.

5.5. Computing infrastructure

Google Colaboratory with NVIDIA A100-SXM4-40GB GPU was used as the computing infrastructure for the deep learning experiments conducted in this paper.

5.6. Code availability

The code is written in PyTorch framework and is publicly available at the project Github repository.

6. Conclusion and further objectives

In this work a neural transducer model with visual attention is presented. It learns to generate LaTeX markup of a real-world math formula given its image. Applying sequence modeling and transduction techniques that have been very successful across modalities such as natural language, image, handwriting, speech and audio; we construct an image-to-markup model that learns to produce syntactically and semantically correct LaTeX markup code.

As future objectives we considers experimenting with different encoder and decoder models in Vision Encoder Decoder architecture. Also we are planning to launch a convenient service for image to LaTeX markup translation for our end user audience.

References

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Belaid, A. and Haton, J.-P. A syntactic approach for handwritten mathematical formula recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (1):105–111, 1984.

Chan, K.-F. and Yeung, D.-Y. Mathematical expression recognition: a survey. *International Journal on Document Analysis and Recognition*, 3:3–15, 2000.

Cireşan, D. C., Meier, U., Gambardella, L. M., and Schmidhuber, J. Deep, big, simple neural nets for handwritten digit recognition. *Neural computation*, 22(12):3207–3220, 2010.

Deng, Y., Kanervisto, A., and Rush, A. M. What you get is what you see: A visual markup decompiler. *arXiv preprint arXiv:1609.04938*, 10:32–37, 2016.

Jaderberg, M., Simonyan, K., Vedaldi, A., and Zisserman, A. Deep structured output learning for unconstrained text recognition. *arXiv preprint arXiv:1412.5903*, 2014.

Karpathy, A. and Fei-Fei, L. Deep visual-semantic alignments for generating image descriptions. arxiv e-prints. *arXiv preprint arXiv:1412.2306*, 2014.

Karpathy, A., Johnson, J., and Fei-Fei, L. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*, 2015.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Luong, M.-T., Pham, H., and Manning, C. D. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

Miller, E. G. and Viola, P. A. Ambiguity and constraint in mathematical expression recognition. In *AAAI/IAAI*, pp. 784–791, 1998.

Niehues, J. and Cho, E. Exploiting linguistic resources for neural machine translation using multi-task learning. *arXiv preprint arXiv:1708.00993*, 2017.

Singh, S. S. `https://untrix.github.io/i2l/140k_download.html`, 2018a. [Online; accessed 10-May-2023].

Singh, S. S. Teaching machines to code: Neural markup generation with visual attention, 2018b.

Sutskever, I., Vinyals, O., and Le, Q. V. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

Wang, T., Wu, D. J., Coates, A., and Ng, A. Y. End-to-end text recognition with convolutional neural networks. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*, pp. 3304–3308. IEEE, 2012.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pp. 2048–2057. PMLR, 2015.

Table 1. Metrics evaluation on training procedure.

Experiment name	Subset of 50 formulas, ViT Decoder	Subset of 500 formulas, ViT Decoder	Full dataset, CNN Decoder
BLEU	70.05	97.65	10.05
ROGUE-1 Precision	0.79	0.97	0.54
ROGUE-1 Recall	0.89	0.99	0.49
ROGUE-1 F-Measure	0.84	0.98	0.51
ROGUE-2 Precision	0.64	0.96	0.13
ROGUE-2 Recall	0.72	0.97	0.12
ROGUE-2 F-Measure	0.68	0.96	0.12
ROGUE-L Precision	0.72	0.97	0.24
ROGUE-L Recall	0.8	0.98	0.22
ROGUE-L F-Measure	0.76	0.97	0.23

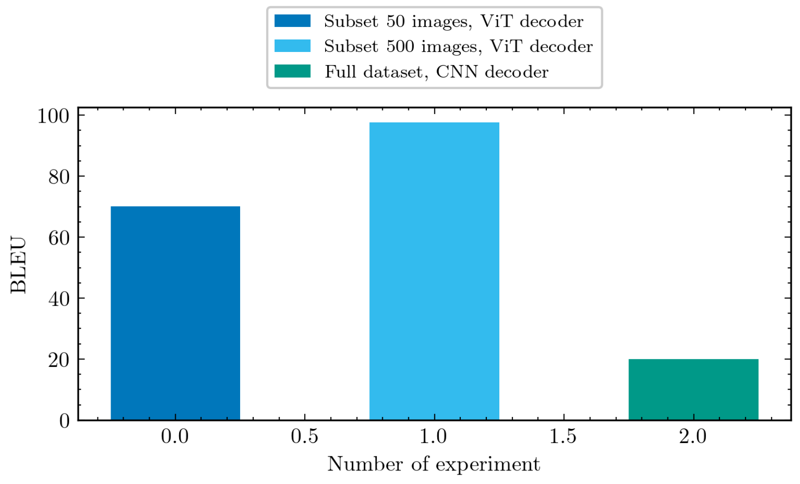


Figure 5. BLEU-metrics evaluation on training procedure.

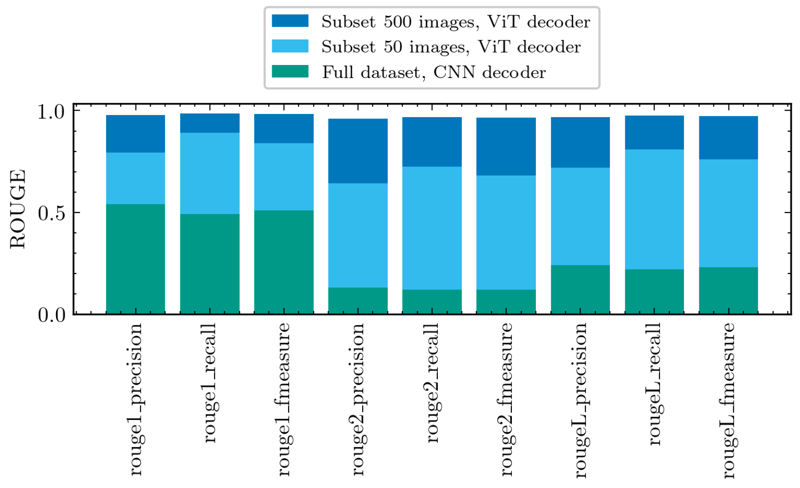


Figure 6. ROGUE-metrics evaluation on training procedure.

Table 2. Metrics evaluation on testing procedure.

EXPERIMENT NAME	FULL DATASET, VIT DECODER
BLEU	50.89
ROGUE-1 PRECISION	0.82
ROGUE-1 RECALL	0.73
ROGUE-1 F-MEASURE	0.77
ROGUE-2 PRECISION	0.44
ROGUE-2 RECALL	0.39
ROGUE-2 F-MEASURE	0.41
ROGUE-L PRECISION	0.62
ROGUE-L RECALL	0.55
ROGUE-L F-MEASURE	0.58

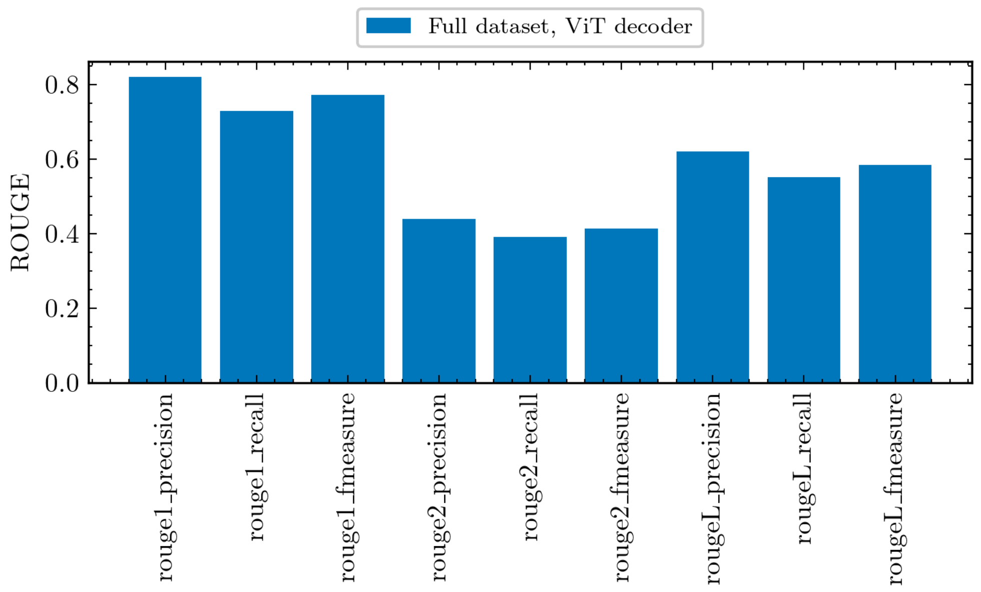


Figure 7. ROGUE-metrics evaluation on testing procedure.

A. Team member's contributions

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

Matvey Skripkin (20% of work)

- Assembling and training the model
- Performing experiments

Nikita Ligostaev (20% of work)

- Preparing the dataset
- Data preprocessing and augmentation
- Dataset and data preprocessing description

Dmitrii Baluev (20% of work)

- Preparing the presentation slides
- Final project speech
- Performing experiments

Nikolay Kalmykov (20% of work)

- Trying CNN-based approach
- Performing experiments

Antonina Kurdyukova (20% of work)

- Project idea inspiration
- Preparing the report
- GitHub repo description