NLP Course Template

Vishnevskiy Alexander

May 2020

Abstract

Link to project code right here:
`https://github.com/alexvishnevskiy/Huawei-project`.

1 Introduction

The problem which I am going to solve is summarization task in Russian. Text summarization is the process of distilling the most important information from a source (or sources) to produce an abridged version for a particular user (or users) and task (or tasks). Nowadays, we have a lot of information and it is important to extract the main idea from a text, in my case the model will help people to generate headlines or summaries for news articles.There are two reasons why Automatic Text Summarization for news summaries is useful:

- Summaries reduce reading time.
- Automatic summarization algorithms are less biased than human summarizers.

The main approaches to this problem are: exctractive and abstractive. Exctractive models select sentences from original text to form summary. In this case all necessary information will be included, but it is not guaranteed that exctract will be coherent and sentences will be connected to each other. However, abstractive models solve this issue by generating new sentences, though it is much harder to train such models. Before transformers state-of-the-art was RNN models that combine exctractive and absctractive approaches [Hsu et al., 2018], but now models such: BART [Lewis et al., 2019], ELECTRA [Clark et al., 2020], PEGASUS [Zhang et al., 2019] beat them. So, my plan is to take BART model, pretrain it on part of RIA news dataset with BPE encoding and fine-tune on generation task.The main advantage of BART is that it is trained by corrupting text with an arbitrary noising function.

1.1 Team

I worked alone, therefore I focused only on one model **BART.**

2 Related Work

ELECTRA [Clark et al., 2020] The authors of this model presented a new way of pretraining. They considered 2 models: discriminator and generator. As for generator they took small LM and masked some of the tokens with MASK symbol, like in BERT. The discriminator in this architecture is actually ELECTRA model, after pre-training they throw out the generator and only fine-tune the discriminator. Important part that the authors mentioned is that the discriminant and the generator trains jointly, however it is different from training GANs. The discriminator is trained with maximum likelihood rather than adversial. Also, if the generator happens to generate the correct token, that token is considered "real" instead of "fake".

$$\mathcal{L}_{gen}(x, \theta_G) = \mathbb{E}\Big(\sum_{i \in m} - \log p_G(x_i | x_{masked}) \Big) \tag{1}$$

$$\mathcal{L}_{dis}(x, \theta_D) = \mathbb{E}\Big(\sum_{t=1}^{n} - \mathbb{1}(x_t^{corrupt} = x_t) \log D(x^{corrupt}, t) \tag{2}$$

$$- \mathbb{1}(x_t^{corrupt} \neq x_t) \log(1 - D(x^{corrupt}, t) \Big)$$

This method is more easier, faster and achieved better results. It achieved 5 GLUE points higher than BERT model. Thefore, this model could be used as extractive model in summarization task instead of BERT.

PEGASUS [Zhang et al., 2019] PEGASUS is pre-trained large Transformer-based encoderdecoder model with a new self-supervised objective. The main idea of pre-training is that important sentences are removed/masked from an input document and are generated together as one output sequence from the remaining sentences. The authors also pointed that this techique works well as a pre-training objective for downstream summarization tasks.

Pointer Generator Network [See et al., 2017] Early methods for abstractive summarization were based on reccurent neural networks. One of the most remarkable was pointer model. Usually RNNs strugle to predict rare or unseen words, however this model could either reproduce the word from recent context or generate it from softmax layer. For each decoder timestep a generation probability is calculated, which weights the probability generating words from the vocabulary, versus copying words from the source text. The vocabulary distribution and the attention distribution are weighted and summed to obtain the final distribution, from which we make our prediction. The distribution over vocabulary is calculated using attention mechanism:

$$e_i^t = v^T \tanh W_h h_i + W_s s_t + b_{attn} \tag{3}$$

$$a^t = softmax(e^t) \tag{4}$$

where v, W_h, W_s, b_{attn} are learnable parameters. The attention distribution is used to produce context vector $h_t = \sum_i a_i^t h_i$, which is fixed-size representation of what has been read, then context vector is concatenated with decoder state and fed through linear layers.$P_{vocab} = softmax(V'(V[s_t, h_t] + b) + b')$, where all parameters except s_t are trainable.The generation probability $p_{gen} = \sigma(w_h^T h_t + w_s^T s_t + w_x^T x_t + b_{ptr})$,where vectors w_h, w_s, w_x and scalar b_{ptr} are learnable parameters and σ is the sigmoid function, is used to switch between vocabulary distribution and attention distribution.Finally, we obtain the distribution over extended vocabulary:

$$P(w) = p_{gen} P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \tag{5}$$

Authors also mentioned that sequence-to-sequence models tend to repeat words, so they adapt coverage vector $c^t = \sum_{t'=0}^{t-1} a^{t'}$, which represents the degree of coverage from words that have been seen. They added this vector to attention mechanism and defined new loss $covloss_t = \sum_i min(a_i^t, c_i^y)$

3 Model Description

BART is autoencoder, which is pretrained on corrupted text. It uses BERT as encoder and GPT as decoder Fig. 1. This model has shown good results on generation tasks, especially summarization, due to the bidirectional encoder, which generalizes text and left to right decoder. Following authors I initialised parameters from $\mathcal{N}(0, 0.02)$ and used GeLu [Hendrycks and Gimpel, 2016] function as activation, which looks similar to ReLu, but more smoothed near 0. I chosed model with 6 encoder layers and 6 decoder layers due to GPU limit in Google Colaboratory.The summarization task can be defined like this: consider we have dataset which consist of texts and corresponding summary. The goal is to find parameters of the model that maximise conditional probability $P(y_t|y_1, \ldots, y_{t-1}, X, \theta)$, where y_t-token at time step t, X-reference text, θ-model parameters, so model parameters can be defined as $\theta = argmax_\theta \prod_i^N P(Y_i|X_i, \theta)$
For pretraining I used RIA news dataset [Gavrilov et al., 2019] and optimized the negative log likelihood of the original document
$\mathcal{L}(\theta) = -\sum_{x \in X} \ln P(y_n|x_n, \theta)$. As authors suggest [Lewis et al., 2019] I used few different noisinig teqniques: token masking(15% of text tokens are randomly masked), token deletion(15% of text tokens are randomly deleted), text infilling(random spans of text are masked, the length of spans are chosen from poisson distribution), sentence permutation.

4 Dataset

I worked with 3 datasets, in order to have better representation of model perfomance.

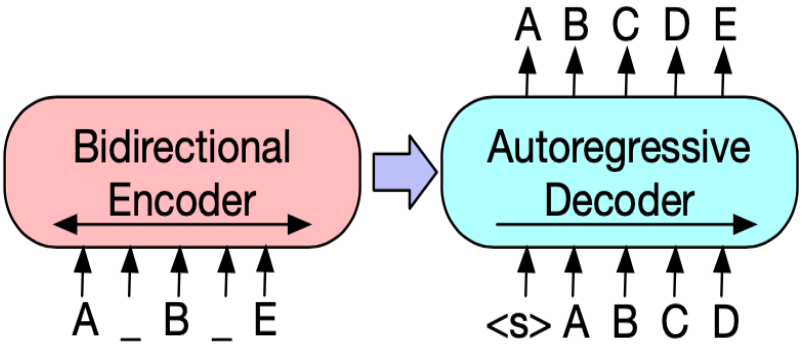

Figure 1: architecture of BART: BERT + Decoder

On the Tab. 4 you can see the statistics for the mentioned datasets used for pretraining and fine-tuning. For pretraining I used 200000 articles of RIO news dataset with propotion 75% for training and 25% for validation. For fine-tuning I used 3 different datasets: RIO news, Svbpress, Gazeta news. RIO news was presented in [Gavrilov et al., 2019], Svbpress news I gathered from their site [1] using «BeautifulSoup4» library, the full code for gathering I provided in my repository , I took Gazeta news dataset from seminar notebook in MIPT. [2]

Pretraining

	Train	Valid
Articles	150000	50000
Tokens	20,088,628	2,017,646
Vocabulary size	30,000	
Out of Vocab rate	0%	

Fine-tuning

Source	Rio			Svbpress			Gazeta		
	Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
Articles	73,963	18,491	5000	22,312	5579	2000	52,400	5265	5770
Tokens	25,769,188	6,438,205	1,847,332	4,400,700	1,097,992	411,171	61,405,676	6,157,595	6,548,405
Vocabulary size	30000								
Out of Vocab rate	0%								

Both for pretraining and fine-tuning I used the same procedure: drop all missing values, drop all NaN values, drop all texts that less than 50 words.

[1] This link

[2] Also datasets for validation: valid , for testing: test

5 Experiments

5.1 Metrics

In my experiements 2 metrics were used: **BLEU** [Wołk and Marasek, 2015] and **ROUGE** [Lin, 2004]. These metrics have shown good results on machine translation task, summarization task, also they highly correlate with human evaluation. BLEU is modified form of precision, the alghoritm can be described like this:

- Compute the modified n-grams counts for all n in all predictions and all their references.(Modified n-gram = for each word in the candidate , the algorithm takes its maximum total count, in any of the references)
- Compute the modified n-grams precisions based on step 1.
- Combine the modified n-grams precisions for all n into a geometric mean.
- Normalize the geometric mean with the brevity penalty.(The main problem of BLEU is that, it prefers short translation, so brevity penalty is added.)

$$\textbf{BLEU} = min(1, \frac{output_{length}}{reference_{length}})(\prod_i precision_i)^{\frac{1}{4}} \tag{6}$$

ROUGE is basically modified recall-oriented metric. Actually, there are few different ROUGE metrics, but I used only 3: ROUGE-1, ROUGE-2, ROUGE-l. ROUGE-1 is overlap of unigrams between predictions and references, ROUGE-2 is overlap of bigrams between predictions and references, ROUGE-l measures longest matching sequence of words. For all these metrics usually calculate recall, precision and F1 score.

$$Recall = \frac{number_{overlappingwords}}{words_{reference}} \tag{7}$$

$$Precision = \frac{number_{overlappingwords}}{words_{system}} \tag{8}$$

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{9}$$

5.2 Experiment Setup

Firstly, I pre-trained BART on part of RIA news dataset with learning rate = $6e^{-4}$, which was found using CycleLR scheduler [Smith and Topin, 2017] with the maximum batch size = 16 available for my GPU. For adjusting learning rate I chose stepLR scheduler with step size = 1 and gamma = 0.7, because it is easily finetuned and still competitive to other schedulers. AdamW with

ϵ=$1e^{-8}$, $weight_{decay}$=0.01 was used as optimizer for both fine-tuning and pre-training. For fine-tuning I chose learning rate = $1e^{-5}$ with the same scheduler and optimizer. All models were trained until convergence. For generation, I adopted beam-search size of 5.

5.3 Baselines

For summarization I used widely-known LEAD-N baseline. The main idea is to take the fisrt n sentences.Even though, this approach looks simple and easy, it is competetive with Neural Networks. In order to compare abstractive method with extractive, I used SummaRuNNer model with BERT pretrained embeddings[3].

6 Results

	BLEU	ROUGE-2	ROUGE-L
Results for GAZETA dataset			
LEAD-N	0.4341	0.1122	0.2264
SummaRunner	0.4145	0.1093	0.2229
BART	0.3851	0.0944	0.2085
Results for RIA dataset			
LEAD-N	0.2025	0.1013	0.1618
BART	0.3866	0.1897	0.3331
Results for Svpressa dataset			
LEAD-N	0.1563	0.0761	0.1488
BART	0.1872	0.0663	0.1692

Table 1: Metrics for all datasets

In Tab. 1 I present results based on three corpora: Svobodnaya Pressa (Svpressa) , the Rossiya Segodnya (RIA) corpus, gazeta.ru (GAZETA).

RIA and GAZETA corpus were significantly larger than svpressa corpora, therefore results for RIA, GAZETA corpora are more reliable and better. For every dataset except GAZETA, BART shows better results than first sentence baseline, however for summaries longer than one sentence it has lower score than abstractive model and lead-n baseline.

In 2 you can see output samples for svpressa and RIA corpora. The model tends to give slightly longer outputs compared to references, but this architecture quite understands big text due to bidirectionality of BERT and gives comprehensive results.

[3]RuBERT model from DeepPavlov

Samples for RIA dataset	
Predicted Summary	**Summary**
большая часть детей, которых граждане сша пытались вывезти из гаити в доминиканской приют, не являются сиротами	большинство детей, которых пытались увезти в сша из гаити, не сироты
тимошенко надеется, что в случае победы на выборах президента будет работать в ее команде премьер-министр украины	луценко будет работать в команде тимошенко, если она победит в выборах
граждане украины решат, кто на ближайшие 5 лет возьмет бразды правления страной в свои руки до 7 февраля	"лютые" выборы: есть ли вероятность второго майдана?
группа людей на воскресенье ворвалась в один из домов сьюдад хуарес в мексиканском заливе еще 20 молодых 13 человек	жертвами бойни на севере мексики стали 13 молодых людей, еще 20 ранены
леверкузенский "байер" вернулся на первое место в чемпионате германии по футболу "фрайбург" 3:1"	"байер" вернулся в лидеры чемпионата германии по футболу
Samples for Svpressa dataset	
Predicted Summary	**Summary**
В Подмосковье попал обокрали ДТП после ДТП в Подмосковье после ДТП в Подмосковье Петербурге В подмосковной Лобне сотрудники эвакуационной службы два человека ВМД после аварии	Сотрудники эвакуационной службы обокрали автомобиль в Подмосковье
В Хабаровском крае убили за убийство кота в жестоком делу по делу-убиймуре к году колонии ЖКР Николаевска-на-Амуре ЖКР В Хабаровском крае	В Хабаровском крае мужчина убил кота и получил год колонии
Путин столичные встречу с визитом в Киргизии в ходе переговоров человека смерть человека Сооронбаем Жээнбековым встречу	Путин прилетел в Киргизию для переговоров с Жээнбековым

Table 2: Output samples.

7 Conclusion

In this paper, I collected dataset, made markup for it, explored the application of BART architecture to the task of abstractive headline generation and summary generation.

References

[Clark et al., 2020] Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators.

[Gavrilov et al., 2019] Gavrilov, D., Kalaidin, P., and Malykh, V. (2019). Self-attentive model for headline generation. In *Proceedings of the 41st European Conference on Information Retrieval*.

[Hendrycks and Gimpel, 2016] Hendrycks, D. and Gimpel, K. (2016). Gaussian error linear units (gelus).

[Hsu et al., 2018] Hsu, W.-T., Lin, C.-K., Lee, M.-Y., Min, K., Tang, J., and Sun, M. (2018). A unified model for extractive and abstractive summarization using inconsistency loss.

[Lewis et al., 2019] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

[Lin, 2004] Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. page 10.

[See et al., 2017] See, A., Liu, P. J., and Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. In *Association of Computational Linguistics (ACL)*.

[Smith and Topin, 2017] Smith, L. N. and Topin, N. (2017). Super-convergence: Very fast training of neural networks using large learning rates.

[Wołk and Marasek, 2015] Wołk, K. and Marasek, K. (2015). Enhanced bilingual evaluation understudy.

[Zhang et al., 2019] Zhang, J., Zhao, Y., Saleh, M., and Liu, P. J. (2019). Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.