**Component-based approach to automatic poetry generation**

Mosolova A. V.

Novosibirsk State University, Novosibirsk, Russia

Bondarenko I. Yu.

Moscow Institute of Physics and Technology, Moscow, Russia

Gusev P. A.

Novosibirsk State University, Novosibirsk, Russia

Malysheva A. D.

Novosibirsk State University, Novosibirsk, Russia

Vodolozasky D. I.

Novosibirsk State University, Novosibirsk, Russia

Borovikova M. N.

Novosibirsk State University, Novosibirsk, Russia

This paper describes two approaches to generating poetic texts on a given topic, one of which we used while participating in ClassicAI, a contest in developing poetry generators in a specific style held by Sberbank. In the first one, we used topic modeling for extracting keywords that a certain topic is characteristic of, applied a text data augmenter to replace parts of the source of the poetry style with thematic words, and then applied a poetic consistency checker to maintain rhyme and rhythm in the output text. In the second one, we used semantic search for obtaining odd lines for the output texts and then phonetic search that selected lines similar in rhyme and rhythm to the given lines and used them as even ones. In this paper, we describe both of our approaches, analyze their benefits and weak spots, provide information on the results of the competition and suggest possible improvements.

Key words: text generation, topic modeling, poetry, embeddings, information retrieval.

**Компонентный подход к автоматической генерации текстов**

Мосолова А. В.

Новосибирский Государственный Университет, Новосибирск, Россия

Бондаренко И. Ю.

Московский Физико-технический Институт, Москва, Россия

Гусев П. А.

Новосибирский Государственный Университет, Новосибирск, Россия

Малышева А. Д.

Новосибирский Государственный Университет, Новосибирск, Россия

Боровикова М. Н.

Новосибирский Государственный Университет, Новосибирск, Россия

1. Introduction

Text generation has been an object of research in natural language processing for quite a long time, and obvious restrictions such as generating a text on a given topic, or in a given style, are often a part of the task. Sberbank of Russia, a Russian bank interested in artificial intelligence, has announced a contest in automatic poetry generation where a competing system is supposed to produce poems that imitate the style of a certain poet (Alexander Pushkin, Sergey Yesenin, Vladimir Mayakovski, Alexander Blok, or Fyodor Tyutchev) to produce poems that are relatable to a certain input (e. g. “logistic regression”); a task, although connected to the tradition of text generation, somewhat more tricky and less obvious. The evaluation was based on the grades assigned to each poem by a human assessors; two qualities of the poem were estimated, its poetic quality and its closeness to the style of the source author.

The approaches used by contestants can be more or less divided into the same two groups that all text generation systems are split into, machine learning based approaches and rule-based component approaches. While text generation based on machine learning is similar in this task and other potential tasks, component approach to creating topic-based style-restricted poetry must apply rather specific rules and is an interesting matter of discussion. In this paper, we present the approaches that we tried, and discuss tasks we had to solve and interesting difficulties that we experienced.

2. Related Work

The field of the automatic poetry generation has been developing for more than half a century. A taxonomy of generative poetry techniques is presented in the paper by C. Lamb et al., 2017[4].

The attempt to generate a text that would match a given topic and style was made in the article by Di Wang et al., 2017 [12]. Their approach used an RNN encoder and an RNN decoder for generating the answer to a given question that would be similar in style to some person.

Another interesting approach to generate poetry on a given topic was introduced in the paper by M. Ghazvininejad et al., 2016[2]. They used a special generated dictionary and rules for detecting word that fit the topic and rhyme. After this, they produced poems by means of a finite-state acceptor and an RNN language model.

An attempt to generate poems on a specific topic was presented in the paper by J. Toivanen et al., 2012[11]. The method in this paper uses two corpora: a semantic corpus and a grammatical corpus. The semantic corpus is presented by a simple word association network and the grammatical corpus which is used for copying structures from the existing poems. The part which creates a poetic structure of the poem wasn’t implemented and fully described in their paper.

The problem of author-stylized poetry was highlighted in the paper by A. Tikhonov and I. Yamshchikov, 2018 [10] in which an LSTM-based model with the use of concatenated word representations was proposed. This solution, evaluated with cross sample

entropy and BLEU metrics, outruns other approaches that also used an LSTM.

3. Component Approaches

In this section we describe two approaches that were implemented and one approach that was another version of the second approach. The algorithm from the parts 3.1.1-3.1.3 was used in Classic AI contest and the algorithms from the parts 3.2.1-3.2.4 we created after the end of the contest.

Before describing our approaches it is necessary to elucidate some terms that will be used in the following parts, namely rhyme and rhythm. Rhyme is any sound repetition that performs the organizing function in the metric composition of a poem [14]. V. Kholshevnikov [3] also states that terminal consonance is mostly used in European poetry and that an exact rhyme i.e. match of sounds of rhyming words starting from the stressed vowel and until the end of the words, dominated in 18-19 centuries. Being guided by this definition, we implemented our approach to checking rhymes. Rhythm is a repetition of any similar phenomenona through commensurable periods [9]; it could occur in a poem by means of counting syllables (syllabic verse), of counting accents (accentual verse) or of stressing and unstressing of vowels (accentual-syllabic verse). A rhythm could be primary (it is related to the decomposition of speech into the intonationally and syntactically complete units) and secondary (this rhythm is connected with the symmetry of combinations of stressed and unstressed vowels or long and short ones). These characteristics of rhythm were also used in our approaches.

3.1. The first approach

This approach is based on topic modeling and a text augmenter for generating new poems and checking rhyme and rhythm by means of an accentor and a transcriptor.

3.1.1. Topic Modeling

As the organizers of the contest stated that possible input topics for the system would be limited to those covered in Wikipedia, we used a dump of Russian Wikipedia to infer 200 topics each of which was connected to a certain vocabulary, i. e. words that are related to this topic. We further extract words that frequently co-occur with items of the said vocabulary and add them to the list of the typical words of the topic. As the component model receives the input topic and the name of the author to imitate, the topic goes to the first component (that of the topic modeling), which automatically picks the closest topic of the 200, extracts the words that are most expected to occur in the generated poem, and these words are passed to the next component.

3.1.2. Text augmenting

Text data augmenter, a program initially developed for augmenting text datasets by replacing words with synonyms [6], appeared useful when turning lists of topic-related words into texts. The augmenting component takes in a corpus of poetic texts with labels of authorship, the name of the author to imitate, the number of texts to generate and imitate, and picks some texts from the corpus. Then, the unit performs morphological tagging on the selected poetic texts, and uses morphological tags to select the words that it can replace without breaking the syntactic structure of the sentence. If, for a given word, there exists a topic word from the previous component that can be inflected into the same form as this word, then the inflected topic word as considered a possible substitution. The unit then combinatorically enlists possible new texts formed by these substitutions to the next unit. The rhythm, the rhyme structure, and the intact words of the original poem help to keep the original style of the author.

3.1.3. Poetry consistency checking

The text augmenting unit yields numerous texts obtained from the original text by replacing some words with topic-related words in the same morphological form. Although this method keeps the syntax structure of the sentence intact, it may break the rhythm structure or the rhymes. This is why we need a separate unit to select those verses that obey the laws of rhythm and rhyme. Presence both rhythm and rhyme are verified by means of an accentor and a transciptor[13]. Texts that were considered not rhythmical or not rhyming were rejected. Two lines were considered rhyming if endings of their transcriptions matched, the ending defined as the part between the last stressed vowel (inclusively) and the end of the line. The output of the system was picked at random out of all acceptable results.

3.2. The second approach

This approach is based on the two types of informational retrieval: semantic search and phonetic search. In the end we describe the attempt to mix this approach and the previous one for obtaining better results.

3.2.1. Semantic search

To come up with a poem topic after the seed sentence (an input topic) was specified, we prepared the dataset in a following way. Firstly, we selected the sentences from the Lenta.ru dataset which were 4 to 6 words long. Then, we applied pre-trained ELMo embeddings [7] to each word in the sentence and used Maximum Pooling to get one embedding for the entire sentence. The resulting sentence embeddings were loaded into the Annoy library, which provides a fast implementation of the Approximate Nearest Neighbors algorithm. After obtaining the seed sentence, we extracted 5 most important words in it by using Rapid Automatic Keyword Extraction algorithm (RAKE) [8], which in comparison to TextRank[5], can work on relatively smaller texts. Next, we construct a seed sentence embedding by using a similar algorithm (ELMo and Maximum Pooling) and extract two nearest neighbors from Annoy. The resulted two sentences will be the first and the third lines in the resulting poem.

3.2.2. Phonetic search

After creating odd lines by means of semantic search we use phonetic search for picking lines that are similar in rhyme and rhythm and take them as even ones. Phonetic search includes sentences sorted by their accents and a ranking model that is based on the method of k-nearest neighbors and these sentences. Each odd line is transformed into a binary vector where 1 stands for a stressed vowel and 0 for an unstressed one. Then this vector is processed by the model and its n closest sentences are produced. These sentences are transformed into phonetic embeddings by means of our model that was trained on poetry transformed from graphemes to phonemes by the transcriptor[13] and divided into a sequence of phonemes. The source sentence (one of the odd lines) is also transformed into a phonetic embedding and the algorithm searches for its k nearest neighbors from n closest embeddings. The result of this algorithm (k closest sentences) is returned as an output (and an input for the next unit of our approach).

3.2.3. Final Ranking

Finally, a special ranking algorithm is applied to choose the most relevant poem for a given topic. We used a regression model trained on grades that the contest’s assessors provided. The training set contained automatically generated poems and related topics with corresponding grades from 1 to 5.

All generated poems and the topic were stemmed, tokenized and then vectorized using character and word n-gram tf-idf vectorizer. Resulting vectors were concatenated into a single vector representation for both poem and topic. Fully-connected neural network with one hidden layer was then used as a regression model. The model evaluates similarity of the poem and the topic estimating it from 1 to 5. Thus, poem with the highest score is chosen among all candidates.

3.2.4. Additional approach

After compiling the results of the previously discussed versions, we decided to implement a version that would combine useful parts of two approaches.

This version consists of semantic search where we replaced ELMo with Fasttext in order to accelerate the generator (because the second version works rather slowly due to using ELMo). The semantic search is used for searching the first and the third line while the second and the forth line are created by the text augmenter. The first (or third) lines are taken and the augmenter produces new lines by replacing all possible nouns, adjectives and adverbs with the words that fit well in the topic (these words are obtained from the topic model described in the section 3.1.1).

4. Results

We presented three approaches to automatic poetry generation. As we participated with one of these systems in the Classic AI, we were able to have one of our algorithms (the one from 3.1 that is based on topic modeling and augmentation) evaluated by the organizers. For evaluation, they used assessors with background in language and literature studies who rated each poem from 1 to 5 on the basis of similarity to the style of a given author and relatedness to a given topic. All participants were ranked by the special metric that takes both of these grades into account. The results of first 5 participants as well as our and the baseline are provided in the figure 1.

As we didn’t have a possibility to ask the assessors that were checking the poems during the competition, we decided to get the philologists from our university to take part in assessing our poems after the end of the competition. They also rated poems from 1 to 5 by style and topic. Unfortunately, the organizers didn’t show a formula that they used for counting the general score so we consider the arithmetic mean between style and topic as the final score. It should be mentioned that we didn’t ask them to rate the poems from the second approach by their style due to the fact that these approaches were created mostly for achieving better results in topic while style wasn’t taken into account and wasn’t required before the beginning of the generation. We suggested them for assessing poems that were generated using our first, second and extended second approaches. They also rated the poems generated by the approach that had the third place in the competition so that we could compare the results not only among our approaches but also using the other competitors’ attempts. The results of this rating are presented in the figure 2. The example for one of the topics with the rate of one of the philologists is presented in the figure 3.

The results of the additional approach outperform the results of the two previous versions by combining their benefits and overcoming their flaws.

Figure 2 Results of Classic AI

|  |  |  |  |
| --- | --- | --- | --- |
| **Team** | **Score** | **Style** | **Topic** |
| Koziev | 0.56585 | 4.47541 | 2.88525 |
| ashvets | 0.50338 | 4.18557 | 2.74227 |
| Markov poems | 0.46960 | 3.48370 | 2.81967 |
| Topspin26 | 0.45689 | 3.79518 | 2.74096 |
| Five Lakes | 0.43696 | 3.90000 | 2.60000 |
| **Our approach** | **0.30550** | **3.24675** | **2.15584** |
| Baseline | 0.11223 | 3.11290 | 1.33871 |

Figure 2 Rating by the philologists

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Score** | **Style** | **Topic** |
| The first approach | 3,6 | 4,14 | 3,1 |
| The second approach | ‒ | ‒ | 3,3 |
| The additional approach | ‒ | ‒ | 3,5 |
| Markov poems | 3,85 | 3,7 | 4 |
| Baseline | 2 | 3 | 1 |

Figure 3 Example of the generation for the topic “Компании, которые не используют искусственный интеллект, в скором времени просто-напросто перестанут существовать на рынке” (A. Block)

|  |  |  |
| --- | --- | --- |
| **Version** | **Poem** | **Style & Topic** |
| The first approach | Сбылось препятствие моё:  перед областной дорогой  ещё однажды тайной силой  погибло препятствие твоё.  И весь исполнен торжества,  я упоён которой тайной  и твёрдо знаю — не случайно  сбывались должные слова. | 4 & 3 |
| The additional approach | Банкноты были искусственно состарены  Политические вопросы не затрагивались  Банкроты плыли искусственно усмотрены  Товаропроизводителей не дотягивались | ‒ & 4 |
| Markov poems | Ей оставались вспоминанья…  Переставай на время плуг Толстого,  Я слишком скоро существовал, дней прошлых слишком много, —  Его отсутствие — пространство мирозданья. | 4 & 5 |

5. Discussion

The idea of our first approach appeared when we participated in ClassicAI with the idea of a neural-based end-to-end approach. While the performance of the latter was rather bad, the first one showed pretty encouraging results. We tried different types of topic modeling but and found LDA most promising. The same attempts were made with different types of representation parts of the source sentence in the augmenter part. Firstly, we used only word-based replacements but after successful results tried using chunks (i.e. noun or verb phrases e.g. “beautiful girl” instead of “beautiful” and “girl”). As for the third part of our first algorithm we only changed the way to estimate the similarity of rhythm and rhyme of the generated poems and of the source ones.

Our second approach that was created only after the end of contest also underwent some changes. At the beginning, our semantic search system used fasttext[1] embeddings instead of ELMo. This approach showed slightly weaker results but worked much faster than the unit with ELMo. That is why we used it for the additional approach. As for the phonetic search, this part was created after considering the fact that semantic search works well so it had itself suggested to try applying this way of creating other lines using another type of embeddings. Later, the idea of mixing appeared so we implemented an augmentation algorithm for generating the even lines. Only the last unit of our second approach stood unchanged from the begging of its implementation for all of the created versions.

We suspect that using new sentence and phonetic embeddings could improve the performance of our model and we will check it in the nearest future.

6. Conclusion

The task suggested by the Classic AI organizers was quite challenging. The first reason for that was the fact that there are no similar systems yet, and the second reason is that the task of poetry generation is extremely unexplored at the current stage of text generation development. In this paper, we presented two approaches to automatic poetry generation. Despite the fact that both were component-based, the first one seems to be more algorithmic whereas the second one is created considering new tendencies in NLP like a wide range of different embeddings and algorithms of machine learning. Consequently, the last approach is also less interpretable than the first one. However, this experience is very useful and we already have the ideas for future improvements of our models. Our approaches are also available on GitHub.

7. Acknowledgements

We greatly appreciate the help of MIPT’s Neural Networks and Deep Learning Lab, National Technology Initiative and PAO Sberbank project ID 0000000007417F630002.

References

1. Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, Vol. 5, pp. 135-146.

2. Ghazvininejad, M., Shi, X., Choi, Y., & Knight, K. (2016), Generating topical poetry, In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, pp. 1183-1191.

3. Kholshevnikov V., (1984), Mysl’, vooruzhennaya rifmami [Мысль, вооруженая рифмами], Saint Petersburg state university, Leningrad.

4. Lamb, C., Brown, D. G., & Clarke, C. L. (2017). A taxonomy of generative poetry techniques, Journal of Mathematics and the Arts, 11(3), pp. 159-179.

5. Mihalcea, R., & Tarau, P. (2004), Textrank: Bringing order into text, In Proceedings of the 2004 conference on empirical methods in natural language processing, Barcelona, pp. 404-411.

6. Mosolova, A., Fomin, V., & Bondarenko, I. (2018), Text Augmentation for Neural Networks, In Supplementary Proceedings of the Seventh International Conference on Analysis of Images, Social Networks and Texts, Moscow, pp. 104-109.

7. Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018), Deep contextualized word representations, In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2018), pp. 2227-2238.

8. Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010), Automatic keyword extraction from individual documents. Text mining: applications and theory, pp. 1-20.

9. Timofeev L., Turaev S. (1974), Slovar’ literaturovedcheskich terminov [Словарь литературоведческих терминов], Prosveschenie.

10. Tikhonov, A., & Yamshchikov, I. P. (2018), Guess who? Multilingual approach for the automated generation of author-stylized poetry, 2018 IEEE Spoken Language Technology Workshop (SLT), Athens, pp. 787-794.

11. Toivanen, J., Toivonen, H., Valitutti, A., & Gross, O. (2012), Corpus-based generation of content and form in poetry, In Proceedings of the third international conference on computational creativity, University College Dublin, pp.175-179.

12. Wang, D., Jojic, N., Brockett, C., & Nyberg, E. (2017), Steering output style and topic in neural response generation, arXiv preprint arXiv:1709.03010.

13. Yakovenko O., Bondarenko I., Borovikova M., Vodolazsky D. (2018), Algorithms for Automatic Accentuation and Transcription of Russian Texts in Speech Recognition Systems, Proceedings of the Speech and Computer - 20th International Conference, SPECOM 2018, pp.768-777.

14. Zhirmunsky V., (1923), Rifma, yeyo istoria i teoria [Рифма, ее история и теория], Russian institute of art history, Petrograd.