

Reprodução Artigo “Educational Question Generation of Children Storybooks via Question Type Distribution Learning and Event-centric Summarization”

Alana Viana Borges da Silva Neo¹

¹Doutoranda em Computação – Universidade Federal de Campina Grande (UFCG)

alana.viana@copin.ufcg.edu.br

Resumo. *O objetivo principal desse artigo é registrar os passos necessários para reproduzir um experimento na construção e utilização de um modelo para gerar questões automaticamente a partir de novos textos de forma automática, técnica chamada de QAG - question-answer generation para a disciplina FPC-CII.*

1. Introdução

O objetivo principal desse experimento foi reproduzir os passos necessários na construção e utilização de um modelo para gerar questões automaticamente a partir de novos textos de forma automática, técnica chamada de QAG - question-answer generation, ou gerador de questões e respostas em uma tradução livre. O modelo utilizado no artigo selecionado é chamado de FairytaleQA e foi treinado com histórias infantis, logo, ele foi utilizado para criar novas questões a partir de uma nova história infantil.

O título do artigo selecionado foi “Educational Question Generation of Children Storybooks via Question Type Distribution Learning and Event-centric Summarization” e o seu código fonte está disponível no github no endereço Educational-Question-Generation de autoria de [Zhao et al. 2022].

O artigo de [Zhao et al. 2022] aborda a geração de questões educativas de contos de fadas com o objetivo de melhorar a capacidade de alfabetização, para gerar perguntas que capturem os aspectos interessantes de uma história de conto de fadas com significado educacional.

No artigo de [Zhao et al. 2022] foi proposto um novo método de geração de perguntas que aprende a distribuição do tipo de pergunta de um parágrafo da história de entrada e depois resume eventos importantes que podem ser usados para gerar perguntas de alta demanda cognitiva. Para treinar o sumarizador centrado em eventos, o modelo de sequência foi ajustado e baseado em um transformador pré-treinado usando amostras compostas por pares de perguntas e respostas educacionais.

Foi utilizada uma base de dados com respostas a perguntas educacionais recém-propostas construído por [Xu et al. 2022]. No conjunto de dados FairytaleQA, foi apresentado um bom desempenho do método de forma automática e com métricas de avaliação humana. São apresentados os três módulos do sistema de geração de perguntas educacionais para livros de histórias, o aprendizado de distribuição do tipo de pergunta, a geração de resumo centrado em eventos e a geração de questões educacionais.

Segundo [Xu et al. 2022], o conjunto de dados FairytaleQA contém arquivos CSV de 278 histórias infantis do Projeto Gutenberg e um conjunto de perguntas e respostas

desenvolvidas por especialistas educacionais com base em uma estrutura teórica baseada em evidências. Este conjunto de dados se concentra na compreensão narrativa dos alunos do jardim de infância e para alunos da oitava série para facilitar a avaliação e o treinamento de habilidades de compreensão narrativa.

Foi tentado reproduzir o experimento discutido no artigo e esse relatório apresenta os principais pontos encontrados.

2. Metodologia

Primeiro tentei reproduzir o artigo de [Zhao et al. 2022]. Ele pede para instalar algumas dependências no python 3.7, mas como pré requisito ele informa para colocar no diretório os dados de treino, mas não especifica como é que deve ser colocado, em quais subdiretórios especificamente. Utilizei uma máquina com Windows 10, com Python 3.7 e Visual Studio Code.

[Zhao et al. 2022] informa que os dados de treino estão em outro repositório, especificamente no repositório disponível no artigo de [Xu et al. 2022]. Como não foi possível reproduzir o experimento de [Zhao et al. 2022], pois ele não especificou corretamente como deveria ser montado o ambiente, mudei os planos e tentei reproduzir o experimento do próprio criador da base de dados [Xu et al. 2022].

Durante a leitura da documentação do artigo de [Xu et al. 2022] existe uma referência a um artigo [Yao et al. 2021]. Esse artigo [Yao et al. 2021] utiliza o conjunto de dados de [Xu et al. 2022] para criar um sistema automatizado de geração de perguntas e respostas, também chamado de QAG System. Esse último trabalho tem uma estrutura e uma documentação melhor do que o original e portanto foi escolhido para reprodução.

3. Reprodução

Seria mais prudente utilizar o Colab e não tentar configurar o ambiente local. O único ponto negativo do Colab é que a sua versão gratuita não tem os recursos computacionais, memória e disco, para o treino do modelo do início ao fim, o que não acontece se for utilizado uma máquina local.

Portanto, para eliminar problemas de recursos, foi utilizado uma máquina com Windows 10 64 Bits com placa de vídeo dedicada (Radeon RX580 Series), o Python 3.7 e como IDE utilizei o Visual Studio Code com o plugin do Jupiter instalado. Além disso, cabe ressaltar que foram utilizadas extensivamente na reprodução do experimento: o Pip que é um package manager para a linguagem Python, o Conda que é um package manager para várias linguagens e o “Anaconda”, um conjunto de módulos para data science com python, que já tem várias bibliotecas configuradas.

Tabela 1. Links dos repositórios

https://github.com/WorkInTheDark/FairytaleQA_Baseline
https://github.com/uci-soe/FairytaleQAData
https://github.com/WorkInTheDark/FairytaleQA_QAG_System
https://github.com/zhaozj89/Educational-Question-Generation

As urls dos repositórios utilizados nessa reprodução estão na Tabela 1.

Figura 1. Instalando Pandas

```
PS C:\alana\FairytaleQA_QAG_System> pip install pandas
Collecting pandas
  Downloading https://files.pythonhosted.org/packages/23/
| 10.6MB 6.4MB/s
```

Figura 2. Instalando Spacy

```
PS C:\alana\FairytaleQA_QAG_System> pip install spacy
Collecting spacy
  Downloading https://files.pythonhosted.org/packages/5f/b9/485d4
100% | 12.0MB 3.0MB/s
```

Todos os projetos da Tabela 1 foram baixados para a máquina local com o github. Em seguida as dependências do projeto foram resolvidas. A primeira dependência a ser solucionada foi o Pandas e o Spacy, que foram instaladas sem maiores dificuldades conforme Figura 1 e Figura 2 no visual studio code conforme as orientações do código fonte.

Figura 3. Instalando Spacy

```
PS C:\alana\FairytaleQA_QAG_System> python -m spacy download en_core_web_sm
Collecting en-core-web-sm==3.3.0 from https://github.com/explosion/spacy-mod
  Downloading https://github.com/explosion/spacy-models/releases/download/en
| 8.7MB 3.3MB/s eta 0:00:02
```

Em seguida foi realizado o download da dependência en_core_web_sm do pacote spacy, conforme Figura 3.

Foi escolhido para reprodução a pesquisa de [Yao et al. 2021]. Nesse trabalho ele treina um modelo com o conjunto de dados FairytaleQA [Xu et al. 2022], esse conjunto de dados foi anotado por vários especialistas relacionados a área de linguística.

Foi copiado o diretório “data-by-train-split” existente do projeto [Xu et al. 2022] para um diretório no projeto [Yao et al. 2021]. Isso era uma necessidade para treinar o modelo. Depois foi alterado o código do projeto [Yao et al. 2021] para informar a existência desse diretório conforme Figura 4 e 5. Além disso existia a necessidade de se manter os arquivos com a extensão question e story no mesmo diretório. Ou seja, foram copiados os arquivos da pasta “data-by-train-split\section-stories\train” para a pasta “data-by-train-split\questions\train” conforme Figura 6.

Depois de configurado os diretórios o projeto de Yao 2021, foi carregado no Visual Studio Code e suas etapas foram reproduzidas. Primeiro foi feito o pré-processamento dos dados originais. Foram 232 arquivos com perguntas e respostas que foram tratados para remoção de espaçamentos e caracteres especiais, conforme Figura 7 e 8. OS métodos estão disponíveis no arquivo “0_Pre_processing_the_original_data.ipynb”

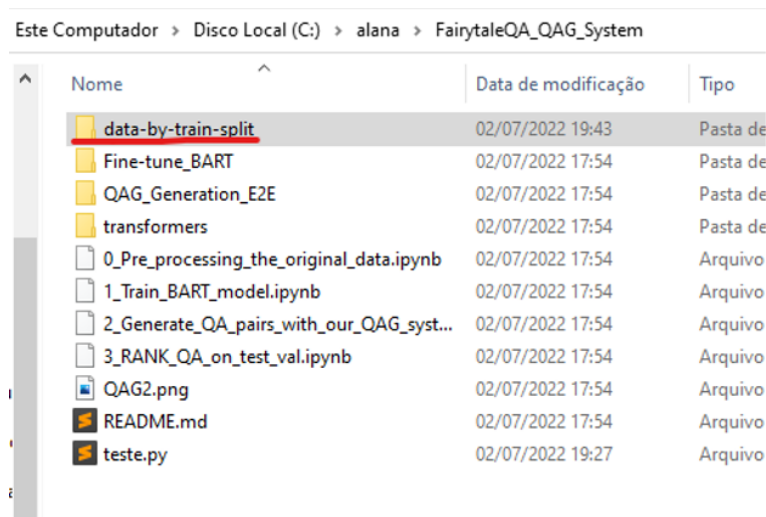
O próximo passo é o treino do modelo que é realizado no arquivo “1_Train_BART_model.ipynb”. As dependências foram instaladas, porém a dependência “faiss” foi comentada pois só funciona no Linux.

Depois do modelo treinado com os dados anotados a partir do “1_Train_BART_model.ipynb” foi a vez de realizar um teste

Figura 4. Configurando os diretórios do projeto

```
dir = "../data-by-train-split/questions/train"
# dir = "../split_for_training/train"
split_version = 'train'
```

Figura 5. Novo diretório criado



Nome	Data de modificação	Tipo
data-by-train-split	02/07/2022 19:43	Pasta de
Fine-tune_BART	02/07/2022 17:54	Pasta de
QAG_Generation_E2E	02/07/2022 17:54	Pasta de
transformers	02/07/2022 17:54	Pasta de
0_Pre_processing_the_original_data.ipynb	02/07/2022 17:54	Arquivo
1_Train_BART_model.ipynb	02/07/2022 17:54	Arquivo
2_Generate_QA_pairs_with_our_QAG_syst...	02/07/2022 17:54	Arquivo
3_RANK_QA_on_test_val.ipynb	02/07/2022 17:54	Arquivo
QAG2.png	02/07/2022 17:54	Arquivo
README.md	02/07/2022 17:54	Arquivo
teste.py	02/07/2022 19:27	Arquivo

“2_Generate_QA_pairs_with_our_QAG_system”. Para isso foi necessário instalar a dependência “allennlp” e “googlecloudstorage”

O modelo treinado também já pode ser baixado em download evitando assim os passos para a sua reprodução a partir do código, o tamanho do modelo treinado é de 4,5 Gigabytes. A esse modelo os autores denominam de “model checkpoint” e tem a extensão ckpt.

Devido a vários ajustes na configuração do ambiente local, uma outra abordagem foi aplicada, tentando realizar a reprodução com o Colab do Google. Porém, quando foi utilizado o colab apareceu uma mensagem informando que o processamento necessário seria maior do que os recursos gratuitos disponíveis pelo Google. Logo em algum momento o uso do colab entraria nessa restrição de dados. Mas os mesmos procedimentos realizados no Visual Studio Code também foram realizados no Colab, conforme Figura 9.

4. Conclusões

A reprodução a partir da construção do modelo preditivo seguiu as etapas de pré-processamento da base de dados original, treino do modelo e sua posterior utilização. A reprodução foi realizada em um ambiente Windows diferente do ambiente original que era um ambiente Linux. Tentar reproduzir o código no colab me pareceu ser uma solução mais viável, dado que o código foi escrito com isso em mente e o problema de dependência provavelmente não iria ocorrer.

A maior parte do tempo foi utilizada para resolver conflitos das bibliotecas do python. Portanto é sugerido certo conhecimento nas bibliotecas: Spacy, transformer, Pytorch, Tensorflow e allennlp.

Figura 6. Pós Pré-processamento dos dados

ste Computador > Disco Local (C:) > alana > FairytaleQA_QAG_System > data-by-train-split > questions > trai

Nome	Data de modificação	Tipo	Tamanho
adventures-of-kintaro-golden-boy-quest...	02/07/2022 16:40	Arquivo CSV	8 KB
adventures-of-kintaro-golden-boy-story....	02/07/2022 16:40	Arquivo CSV	17 KB
a-fish-story-questions.csv	02/07/2022 16:40	Arquivo CSV	3 KB
a-fish-story-story.csv	02/07/2022 16:40	Arquivo CSV	5 KB
a-french-puck-questions.csv	02/07/2022 16:40	Arquivo CSV	4 KB
a-french-puck-story.csv	02/07/2022 16:40	Arquivo CSV	6 KB
a-legend-of-confucius-questions.csv	02/07/2022 16:40	Arquivo CSV	4 KB
a-legend-of-confucius-story.csv	02/07/2022 16:40	Arquivo CSV	3 KB
a-legend-of-knockmany-questions.csv	02/07/2022 16:40	Arquivo CSV	11 KB
a-legend-of-knockmany-story.csv	02/07/2022 16:40	Arquivo CSV	22 KB
ali-baba-and-forty-thieves-questions.csv	02/07/2022 16:40	Arquivo CSV	3 KB
ali-baba-and-forty-thieves-story.csv	02/07/2022 16:40	Arquivo CSV	5 KB
a-lost-paradise-questions.csv	02/07/2022 16:40	Arquivo CSV	5 KB
a-lost-paradise-story.csv	02/07/2022 16:40	Arquivo CSV	8 KB
anent-giant-who-did-not-have-his-heart...	02/07/2022 16:40	Arquivo CSV	8 KB
anent-giant-who-did-not-have-his-heart...	02/07/2022 16:40	Arquivo CSV	13 KB
aspenclog-questions.csv	02/07/2022 16:40	Arquivo CSV	3 KB

Figura 7. Arquivos tratados

```
[12] ✓ 0.1s
...
====
232 232
cnt_files: 464
# of books: 232
```

Ter dados anotados para a construção do modelo viabilizou ao menos 4 pesquisas publicadas a partir do mesmo trabalho, ter uma equipe organizada ao redor dessa base de dados trouxe ganhos para as possíveis soluções para o problema.

É cada vez mais importante ter código revisado, sem comentários e bem escrito, pois a maior parte do tempo gasto na reprodução se deu por pequenos detalhes no código e não pela complexidade da arquitetura proposta.

A disponibilidade do modelo já treinado e do código para o treino do modelo garante a reprodutibilidade do experimento, o que não aconteceria se o código e os dados de treino não fossem disponibilizados.

Referências

- Xu, Y., Wang, D., Yu, M., Ritchie, D., Yao, B., Wu, T., Zhang, Z., Li, T. J.-J., Bradford, N., Sun, B., Hoang, T. B., Sang, Y., Hou, Y., Ma, X., Yang, D., Peng, N., Yu, Z., and Warschauer, M. (2022). Fantastic Questions and Where to Find Them: FairytaleQA – An Authentic Dataset for Narrative Comprehension.
- Yao, B., Wang, D., Wu, T., Zhang, Z., Li, T. J.-J., Yu, M., and Xu, Y. (2021). It is AI's Turn to Ask Humans a Question: Question-Answer Pair Generation for Children's Story Books.
- Zhao, Z., Hou, Y., Wang, D., Yu, M., Liu, C., and Ma, X. (2022). Educational Question Generation of Children Storybooks via Question Type Distribution Learning and Event-Centric Summarization.

Figura 8. Geração das perguntas e respostas

```
print("# stories: ", cnt)
print("# question: ", cnt_q)
```

✓ 3m 0.9s

Output exceeds the [size limit](#). Open the full output data [in a text editor](#)

a-fish-story
./data-by-train-split/questions/train/a-fish-story-story.csv
a-french-puck
./data-by-train-split/questions/train/a-french-puck-story.csv
a-legend-of-confucius

Figura 9. Treino do Modelo

