Comparação entre BART e Pointer-Generator Networks: Sumarização de Texto com Restriçõs de Hardware e Tempo

Marcus Vinícius Reisdoefer Pereira
mvrp21@inf.ufpr.br
Departamento de Informática
Universidade Federal do Paraná
Curitiba, Paraná, Brasil

Resumo

Este trabalho tem como objetivo replicar os resultados dos experimentos realizados no artigo "Get To The Point: Summarization with Pointer-Generator Networks" [4], utilizando a arquitetura de *transformers BART* [2], um modelo de estado da arte em tarefas de processamento de linguagem natural. A proposta era investigar se o *BART* poderia atingir ou superar as *Pointer-Generator Networks* na tarefa de sumarização de texto, avaliando o desempenho através das métricas de ROUGE em um ambiente de *hardware* comum (direcionado a consumidores comuns), com tempo de treinamento limitado. Apesar de todos os esforços, não foi possível alcançar os *scores* de ROUGE reportados no estudo original, possivelmente devido às limitações computacionais e ao tempo restrito de treinamento, sugerindo que fatores como a escala de hardware e a quantidade de recursos computacionais disponíveis são cruciais para a obtenção de desempenho ideal. Também é possível que o desempenho abaixo das expectativas neste experimento esteja relacionado a limitações na aplicação de certos métodos ou técnicas, o que pode ser atribuído a lacunas na experiência ou na familiaridade com os procedimentos adotados. Tais desafios operacionais podem ter impactado negativamente os resultados, indicando que a falta de habilidades específicas em determinadas etapas do processo contribuiu para o desfecho observado. Este trabalho, portanto, oferece *insights* sobre as dificuldades práticas ao se replicar experimentos com modelos avançados em configurações mais limitadas.

Palavras-chave

Processamento de Linguagem Natural, Sumarização de Texto, BART, Pointer-Generator Networks

1 Introdução

A sumarização de texto tem sido uma tarefa central em processamento de linguagem natural (PLN), com aplicações em diversos domínios, como jornalismo, suporte ao cliente e sistemas de recomendação. O objetivo principal dessa tarefa é gerar resumos concisos e informativos de textos longos, mantendo as informações essenciais, o que é particularmente desafiador em textos complexos e extensos. Existem duas abordagens principais para a sumarização de texto: a sumarização abstrativa e a extrativa. A primeira visa gerar novos enunciados que capturam a essência do texto original, enquanto a segunda seleciona diretamente segmentos do texto para formar o resumo.

Entre alternativas abstrativas, destaca-se o trabalho de See et al. (2017), que propôs um modelo de sumarização sequencial baseado em redes neurais com =atenção, conhecido como *Pointer-Generator Network*, para resolver problemas de gerar resumos abstrativos de alta qualidade. Esse modelo, ao combinar uma rede geradora com um mecanismo de *pointer*, apresentou resultados notáveis, especialmente em termos de fidelidade ao conteúdo original. O modelo alcançou um ROUGE-1 de 39.53, que foi o melhor desempenho para a tarefa de sumarização na época.

Todavia, já sabe-se que utilizando *transformers* é possível não só alcançar mas também superar os resultados obtidos em 2017, o que foi feito utilizando BERT reduzindo o problema de sumarização para um problema de *text matching*, abordando o problema de sumarização extrativa, que obteve ROUGE-1 44.41, superando os 39.53 do *Pointer Generator* ainda em 2020 [5].

Enquanto o modelo proposto por See et al. (2017) foi uma grande contribuição para a área de sumarização, observa-se na publicação feita que o tempo de treinamento para a rede proposta levou dias [4], e utilizou de uma *GPU NVIDIA Tesla K40m*, que não é comum para a maioria dos computadores pessoais. Portanto esse *paper* se dedica a estudar se é possível alcançar os resultados do artigo original [4] de 2017 utilizando as técnicas conhecidas do estado da arte de 2024, em particular *transformers*, mas introduzindo limitações de tempo e utilizando *hardware* de consumidor comum.

2 Escolha de modelo

A escolha do modelo *BART-small* para a tarefa de sumarização de texto neste estudo foi motivada por uma combinação de fatores técnicos e práticos. *BART* (*Bidirectional and Auto-Regressive Transformers*) foi desenvolvido como um modelo de *seq2seq* altamente eficiente para tarefas de geração de texto como a sumarização abstrativa.

Outro ponto importante é a flexibilidade e eficácia do *BART* em várias tarefas de NLP. O modelo tem demonstrado resultados excepcionais em uma série de benchmarks, incluindo sumarização, tradução e resposta a perguntas, o que o torna uma escolha natural para este estudo. O desempenho robusto do *BART* em tarefas de geração de texto, aliado à sua facilidade de adaptação e capacidade de lidar com diferentes tipos de entrada, é uma das razões pelas quais decidimos utilizá-lo na comparação com a análise original [4].

Além disso, a escolha pela versão *BART-small* em vez de modelos maiores, como o *BART-base* ou o *BART-large*, foi uma decisão pragmática baseada nas limitações de *hardware* auto-impostas para o experimento. Modelos como o *BART-large*, embora frequentemente mais poderosos, exigem significativamente mais memória e poder computacional para o treinamento e a inferência, o que dificultaria a execução do experimento em sistemas com recursos limitados. O *BART-small*, por sua vez, oferece uma boa *trade-off* entre desempenho e custo computacional, mantendo uma arquitetura

suficientemente poderosa para tarefas como a sumarização, sem exigir mais do que os recursos disponíveis.

Portanto, a escolha do modelo *BART-small* neste estudo foi fundamentada em sua eficiência, flexibilidade e balanceamento entre desempenho e requisitos computacionais, além de sua comprovada eficácia em tarefas de sumarização de texto. Essas características tornam o *BART-small* uma escolha atraente para explorar o potencial de modelos de transformadores no contexto de hardware comum, onde as restrições de memória e tempo de treinamento são fatores críticos.

3 Método

Como o objetivo desse trabalho é verificar se é possível atingir resultados de 2017 em 2024, utilizando técnicas conhecidas atualmente **mas com limitações** técnicas de *hardware* e tempo, decidiu-se utilizar uma *RTX 4060 Ti* como *GPU* para os testes, por ser uma "*GPU* de consumidor", e impôs-se uma limitação de aproximadamente 3 horas de treino (ou *fine-tuning*) da rede para a sumarização de textos. O código dos experimentos foi deixado acessível publicamente [3], para fins de apresentação.

Para comparação com o modelo original [4], serão utilizadas as métricas quantitativas ROUGE-1, ROUGE-2 e ROUGE-L. O ROUGE-1 mede a sobreposição de unigramas entre o resumo gerado e o resumo de referência, avaliando a capacidade do modelo em capturar termos individuais presentes no texto original, refletindo a precisão e a cobertura em termos de palavras isoladas. O ROUGE-2, por sua vez, avalia a sobreposição de bigramas, sendo mais sensível à estrutura gramatical e semântica do texto, o que permite verificar a habilidade do modelo em capturar pares de palavras que ocorrem juntas no conteúdo original. Já o ROUGE-L mede a maior subsequência comum entre o resumo gerado e o resumo de referência, levando em consideração a ordem das palavras (essa métrica é especialmente relevante para avaliar a fluência do texto gerado, pois considera a coesão e a sequência das palavras, o que é essencial para sumarizações abstrativas que precisam preservar o sentido e a coerência do conteúdo original).

Essas três métricas são amplamente utilizadas para avaliar a qualidade de modelos de sumarização, pois fornecem uma medida objetiva da similaridade entre o resumo gerado e o texto de referência sem ser uma mera comparação de *strings*, considerando tanto a cobertura de informações quanto a preservação da estrutura e fluência.

A definição dos valores para *batch size* e *learning rate* foram feitas empiricamente, por meio de observações do comportamento de tempo da execução do código, assim como o comportamento da função de *loss* do modelo e do uso de memória da *GPU*. Ainda, o número de épocas foi fixo em 3, pois observou-se que um maior número de épocas começava a causar *overfitting* do modelo, e a *loss* não se reduzia de maneira significativa, muitas vezes inclusive aumentava.

Como *dataset* para os experimentos foi utilizado o mesmo que o artigo original [4], o CNN Daily Mail, obtido do *website HuggingFace* [1]. Como foi imposta uma limitação de tempo de treino da rede proposta nesse trabalho, apenas um subconjunto do *dataset* de treino foi utilizado (65536 de 287113), que foi o número de itens que possibilitou um tempo de treinamento de aproximadamente 3 horas, em comparação com o total de mais de 13 horas do *dataset* inteiro.

Apenas por questão de curiosidade foram feitos dois experimentos treinando a rede com o *dataset* inteiro, ignorando a limitação de tempo, e não foram observadas mudanças significativas nos *scores* ROUGE-*, o que indica que os desafios encontrados não estão diretamente relacionados ao tamanho do *dataset* de treino.

4 Resultados e Conclusões

Após alguns experimentos breves observando o comportamento do modelo executando com diversos parâmetros, foram obtidos valores finais para a obtenção dos resultados desejados. Os parâmetros finais estão disponíveis na tabela 1.

Parâmetro	Valor
Modelo	BART-small
GPU	NVIDIA RTX 4060 Ti (8GB)
Batch Size	8
Épocas	3
Tamanho dataset de treino	65536 (2^{16})

Tabela 1: Configurações finais para os experimentos

Utilizando essas configurações o *fine-tuning* do modelo levou aproximadamente duas horas e meia. Os resultados para as métricas ROUGE-* estão disponíveis na tabela 2.

Rede	ROUGE-1	ROUGE-2	ROUGE-L
BART-small limitado	23.61	11.65	19.74
Poiter-Generator	39.53	17.28	36.38

Tabela 2: Métricas ROUGE obtidas.

Observa-se que todas as métricas foram bem inferiores às obtidas pelo *Pointer-Generator*, sendo que a mais próxima foi a ROUGE-2, que também foi inferior por uma grande margem.

Portanto, conclui-se que mesmo utilizando técnicas mais avançadas do estado da arte atual de *Deep Learning* ainda é inviável alcançar os resultados de técnicas mais antigas, se limitações grandes de *hardware* e tempo estiverem presentes. Verifica-se que os resultados obtidos com as limitações impostas pelo experimento mal são comparáveis às técnicas do estado da arte de 2015, que apresentaram valores de ROUGE-* próximos, mas maiores, aos observados pelo modelo atual.

Nota-se que por mais que o modelo proposto nesse trabalho não alcançou os resultados esperados em termos de métricas, as sumarizações feitas pelo modelo, quando empiricamente observadas, estavam compreensíveis e boa parte continha a maior parte das informações do texto original. Sendo assim, por mais que não seja um modelo tão poderoso quanto o *Pointer-Generator* ou *transformers* mais recentes, ainda é uma opção viável para experimentação pessoal.

Agradecimentos

Agradeço ao professor da disciplina de *Deep Learning*, Paulo R. Lisboa de Almeida, pela oportunidade de realizar essa pesquisa.

Também gostaria de reconhecer os trabalhos de See et al. (2017) e Zhong et al. (2020), cujas contribuições foram fundamentais para o desenvolvimento deste estudo, especialmente no que diz respeito à inspiração e fundamentação para a abordagem adotada. Agradeço ainda a todos que, de alguma forma, contribuíram para a realização deste trabalho.

Referências

[1] Hugging Face. 2020. CNN/Daily Mail Dataset. https://huggingface.co/datasets/cnn_dailymail Accessed: 2024-11-10.

[2] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. *CoRR* abs/1910.13461 (2019). arXiv:1910.13461 http://arxiv.org/abs/1910.13461

[3] Marcus V. Reisdoefer Pereira. 2024. DeepLearning-T1. https://github.com/mvrp21/DeepLearning-T1 Accessed: 2024-11-17.

[4] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. *CoRR* abs/1704.04368 (2017). arXiv:1704.04368 http://arxiv.org/abs/1704.04368

[5] Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive Summarization as Text Matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 6197–6208. https://doi.org/10.18653/v1/2020.acl-main.552