Uso de Processamento de Texto Para Geração de Legendas a Partir de Imagens de Gráfico

Christian S. de Lira[1], Giuseppe F. Neto[1], Jonas Freire[1], Wilder Carvalho[1]

[1]Departamento de Computação – Universidade Federal Rural de Pernambuco (UFRPE)

{christian.lira, giuseppe.fiorentinoneto, jonas.freire, wilder.carvalho}@ufrpe.br

Abstract. *Charts are tools that facilitate the analysis and interpretation of a data set, being fundamental in scientific articles and other media, facilitating the understanding of data and numerical values. Thus, this study proposes the use of word processing techniques along with deep machine learning technologies to generate line graph image captions. The goal is to extract information from the graph image and thus be able to generate a new appropriate caption from a new entry. The result of the study shows that the methodology adopted, even with a small database have interesting results, even the evaluation metric indicating bad results, and that should be further investigated.*

Resumo. *Gráficos são ferramentas que facilitam a análise e interpretação de um conjunto de dados, sendo fundamental em artigos científicos e outros meios, facilitando a compreensão dos dados e valores numéricos. Assim, este estudo propõe o uso de técnicas de processamento de texto junto com tecnologias de aprendizagem de máquina profunda para gerar de legendas de imagens de gráficos de linha. O objetivo é extrair informações da imagem do gráfico e com isso poder gerar, a partir de uma nova entrada, uma nova legenda apropriada. O resultado do estudo mostra que a metodologia adotada, mesmo com uma base de dados pequena possuem resultados interessantes, mesmo com métricas de avaliação que indicam péssimos resultados, e que devem ser melhor investigados.*

1. Introdução

No ramo da inteligência artificial, o uso de Processamento de Linguagem Natural (PLN), possibilita a Sumarização Automática de Textos, que torna viável a recuperação de informações utilizando técnicas e algoritmos para identificar e coletar, ou gerar, sentenças a partir de documentos textuais [CABRAL et al. 2018].

Sendo assim, encontra-se facilmente inúmeras abordagens de sumarização de documentos de texto, inclusive sumarização de artigos científicos. No entanto, artigos científicos não se utilizam apenas de ferramentas textuais, mas também de ferramentas visuais para o apoio e comprovação do argumento apresentado.

Consequentemente, entende-se que gráficos são ferramentas fundamentais em artigos científicos, uma vez que, não só comprovam o argumento do autor e resume os resultados, como também torna o artigo mais legível, evitando o uso de grandes tabelas de dados de difícil interpretação.

Deste modo, a extração de informações de gráficos serve de auxílio para a melhora dos resultados obtidos na sumarização de documentos usando técnicas de PLN. Todavia, PLN não pode ser usada diretamente para extrair informações de imagens gráficas.

Portanto, este estudo propõe o uso de técnicas de processamento de texto em conjunto com uma Rede Neural Recorrente para geração de legendas a partir de imagens de gráficos de linha.

2. Metodologia

O uso de uma boa metodologia nos traz bons resultados. Quando bem estruturada e eficaz pode ser decisiva no resultado final. Para isso, fizemos uso de técnicas atuais e uma construção dos dados de forma ideal.

Primeiramente temos a base de dados, que foi criada com auxílio manual. Por se tratar de imagens de gráficos, um pré-processamento é realizado. O processamento de texto é feito para uma melhoria no vocabulário, este que será utilizado numa rede neural.

Por fim, temos o modelo de rede neural utilizado e uma avaliação de como foi o desempenho da mesma demonstrando os prós e contras de ter utilizado tal metodologia.

2.1. Base de dados

Tendo em vista que, o foco do estudo foi a extração de características de imagens de gráficos de linha, foi criada, utilizando a linguagem de programação Python em conjunto com a biblioteca Matplotlib [Hunter 2007], uma base de dados com 200 gráficos de linha, com informações aleatórias. Além disso, criamos legendas para todos os gráficos gerados, com o objetivo de auxiliar a aprendizagem supervisionada da rede neural que será utilizada em conjunto com as características extraídas da imagem, para gerar as legendas dos gráficos.

2.2. Extração de características da imagem

Primeiramente, com a ideia de ter informações a respeito das imagens, fizemos uma etapa de pré processamento. Para isso, teremos a extração de características da imagem, a qual será responsável pela obtenção de *features* da imagem.

Para a realização da extração de características utilizamos o modelo pré-treinado, do *Keras Applications* [Chollet et al. 2015], de redes convolucionais profundas, o VGG19: um modelo de rede neural que possui 19 camadas. Tal modelo refere-se a uma profunda rede convolucional para reconhecimento de objetos, desenvolvida e treinada pelo *Visual Geometry Group (VGG)* de Oxford [Simonyan and Zisserman 2014], sendo específica para extração de *features* de imagens, o que nos ajuda bastante para o problema em questão.

Inicialmente cada imagem é passada para o método que irá realizar a extração de características. Dentro do método de extração, algumas operações de redimensionamento são feitas para que esta fique no padrão aceitável pela rede. Após os procedimentos, passamos a imagem pra rede neural que irá realizar a extração de características e retornar as características mais convenientes para o mesmo.

Realizada a extração, salvamos o seu resultado em um arquivo, para, no futuro, podermos realizar o mapeamento das legendas com suas características.

2.3. Processamento de texto

Algumas etapas de processamento de texto foram realizadas em cima das descrições de cada imagem da base de dados. O processamento textual em cima das descrições, segue os passos: conversão de todas as palavras em minúsculas, aplicação do *stemming* para obtenção do radical das palavras, tendo em vista que o objetivo é a geração texto para outros gráficos. Após a aplicação do *stemming*, remove-se os *stop words*, tais como pontuação, palavras com um caractere, e etc. Em seguida remove-se os acentos das palavras.

Após todas as etapas de processamento de texto, é gerado um vocabulário de palavras para o texto obtido. O vocabulário de palavras trata-se de um método que guarda uma única instância de todas as palavras contidas no texto, isto é, não há repetição de palavras.

Depois dessas etapas, as descrições são salvas em um arquivo de texto para uso posterior das mesmas na rede neural.

2.4. Modelo de rede profunda

2.4.1. Preparando os dados para a rede

Utilizamos os dados processados e devidamente tratados, como explicados na seção anterior, para que possamos ajustá-los ao modelo. Os dados de treino serão todas as imagens e legendas no conjunto de dados de treinamento. O conjunto de dados de treinamento e teste é selecionado aleatoriamente na proporção 80% de treino e 20% de teste. Este conjunto contém as listas de nomes de arquivos de imagens e a partir desses nomes de arquivo, podemos extrair os identificadores de imagens e usá-los para filtrar imagens e descrições para cada conjunto.

O modelo de rede profunda utilizada possui um ramo, que pode ser visto na figura 1, que irá gerar uma legenda dada uma imagem. A geração da legenda se dá pela geração de uma palavra de cada vez utilizando um algoritmo de Rede Neural Recorrente (RNN), visto na figura 2, que supera alguns dos problemas das RNNs convencionais, chamada LSTM ou Long Short Term Memory, cujo modelo é resumido na figura 3. Este é um tipo de rede neural recorrente, que é usada em diversos cenários de Processamento de Linguagem Natural. Como a sequência de palavras geradas anteriormente será fornecida como entrada precisaremos de uma 'primeira palavra' para iniciar o processo de geração e uma 'última palavra' para sinalizar o final da legenda.

Assim usaremos as sequências de caracteres 'begineq' e 'endseq' para esse fim. Esses *tokens* são adicionados às descrições carregadas à medida que são carregadas. Com isso precisamos apenas passar o token de descrição inicial 'begineq' e a rede irá gerar uma palavra e em seguida chamaremos o modelo recursivamente com as palavras geradas como entrada até o final do token de sequência ser atingido 'endseq' ou atingir o tamanho máximo da descrição.

O texto da legenda precisará ser codificado para números antes de poder ser apresentado ao modelo como entrada ou comparado às previsões do modelo. Assim foi feito um mapeamento de palavras para valores inteiros exclusivos, que foi feito utilizando a biblioteca Keras que fornece a classe *tokenizer* que pode aprender esse mapeamento a partir dos dados de descrição carregados.

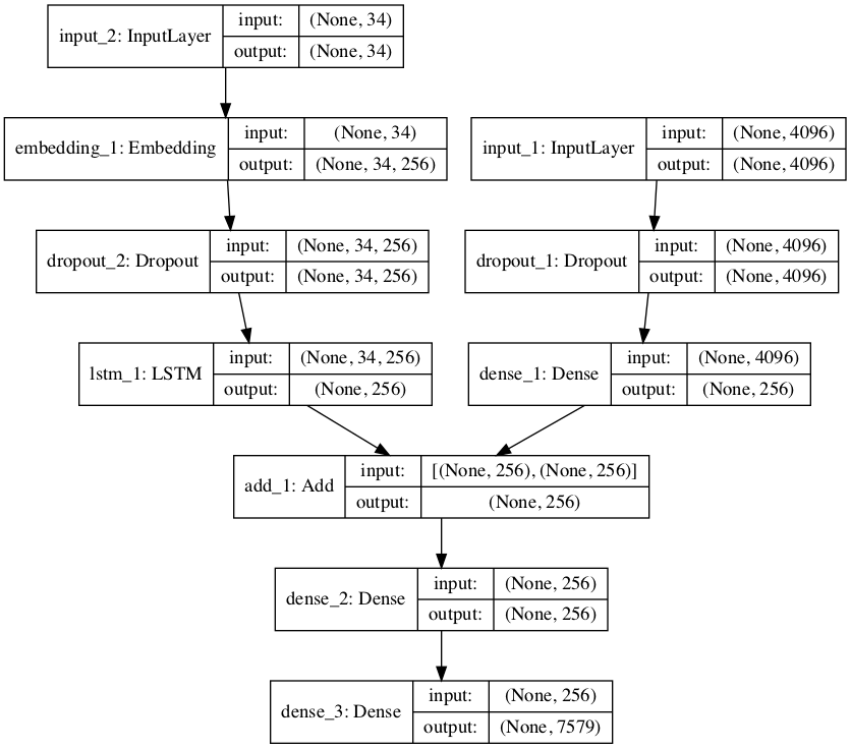

Figura 1. Modelo de aprendizado profundo da geração de legendas

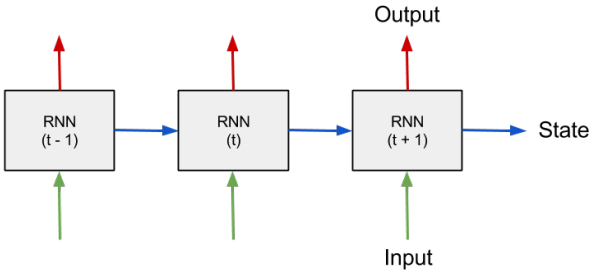

Figura 2. Modelo de Rede Neural Recorrente

Cada descrição será dividida em palavras. O modelo receberá uma palavra e a imagem e gerará a próxima palavra. Em seguida, as duas primeiras palavras da descrição serão fornecidas ao modelo como entrada com a imagem para gerar a próxima palavra. É assim que o modelo será treinado. Posteriormente, quando o modelo for usado para gerar descrições, as palavras geradas serão concatenadas e fornecidas recursivamente como entrada para gerar uma legenda para uma imagem.

2.4.2. Definindo o modelo

Existem duas matrizes de entrada no modelo: uma para os recursos da imagem e outra para o texto codificado. Há uma saída para o modelo, que é a próxima palavra codificada na sequência de texto. O texto de entrada é codificado como números inteiros, que serão alimentados em uma camada de incorporação de palavras. Os recursos da imagem

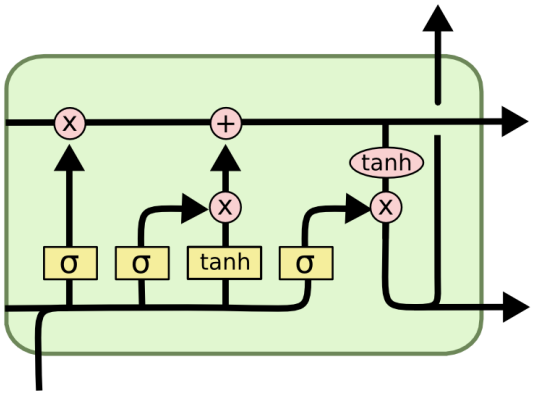

Figura 3. Modelo de Rede Neural LSTM

serão alimentados diretamente para outra parte do modelo. O modelo produzirá uma previsão, que será uma distribuição de probabilidade sobre todas as palavras do vocabulário. Como pode ser visto na figura 1.

Os dados de saída serão, portanto, uma versão codificada do *one-hot encoded* para cada palavra, representando uma distribuição de probabilidade idealizada com valores 0 em todas as posições de palavras, exceto na posição atual da palavra, que tem o valor 1.

Com essa ideia utilizaremos o modelo de aprendizado profundo com base no ”modelo de mesclagem”[Tanti et al. 2017]. Como descrito no artigo ”Caption Generation with the Inject and Merge Encoder-Decoder Models” [Jason Brownlee 2017] o autor fornece um modelo que segue o esquema reproduzido na figura 4.

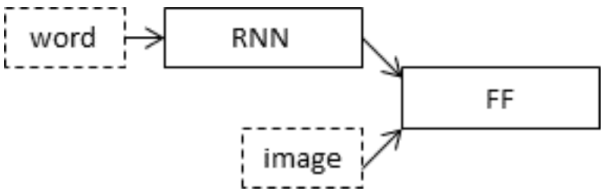

Figura 4. Modelo desenvolvido por *Marc Tanti, et al*

2.5. Avaliação do modelo de legenda

Após ter definido o modelo, foi possível estudar uma forma de avaliar a habilidade do modelo no conjunto de dados de teste de validação. Esta avaliação foi feita gerando descrições para todas as imagens no conjunto de dados de teste e avaliando essas previsões com uma função de custo padrão. As descrições reais e previstas foram coletadas e avaliadas coletivamente usando a pontuação do corpus *BLEU* que resume o quão perto o texto gerado está do texto esperado

As pontuações BLEU são usadas na tradução de texto para avaliar o texto traduzido em uma ou mais traduções de referência. A biblioteca NLTK Python implementa ou calcula a pontuação BLEU na função *corpus_bleu*.

2.5.1. Estudo de Avaliação BiLingual - BLEU

Bi-Lingual Evaluation Understudy (BLEU), em português Estudo de Avaliação BiLingual, é uma métrica de avaliação automática do texto **traduzido** por máquina. A pontuação BLEU é um número entre zero e um que compara a similaridade do texto traduzido por máquina com um conjunto de traduções de referência de alta qualidade. O valor 0 significa que a saída da tradução de máquina não coincide com a tradução de referência (baixa qualidade). O valor 1 significa perfeita correspondência com as traduções de referência (alta qualidade).

Na figura 5 é possível verificar como o AutoML do google, que fornece modelos de ML em apenas um clique, mantendo toda a implementação do código longe do usuário, representa os resultados do BLEU

Pontuação BLEU	Interpretação
< 10	Praticamente inútil
10 - 19	Difícil de compreender o sentido
20 - 29	O sentido está claro, mas há erros gramaticais graves
30 - 40	Pode ser entendido como boas traduções
40 - 50	Traduções de alta qualidade
50 - 60	Traduções de qualidade muito alta, adequadas e fluentes
> 60	Em geral, qualidade superior à humana

Figura 5. Interpretação das pontuações BLEU

3. Resultados

3.1. Processamento das imagens

Após alimentar como entrada as imagens da base de dados, a rede VGG19 nos retorna as *features* que foram consideradas mais relevantes. Com isso podemos mapear os 200 nomes das imagens para cada uma das características, de dimensão 1x4096, extraídas pela rede, para que se possa alimentar o modelo principal definido na seção 2.4.2.

3.2. Processamento de texto

Foram carregados 200 legendas, com tamanhos variados indo de 117 até 659 número palavras em uma legenda, de um arquivo, que estava estruturado como dito na seção 2.3, desse arquivo apenas um vocabulário de tamanho 1266 foi salvo, pois indica o conjunto de palavras únicas. Para isto foram feitas todas as reduções, pois um vocabulário maior, com muitas palavras podem gerar erros de ortografia ou serem usadas apenas uma vez em todo o conjunto de dados. Assim o refinamento dos textos em um vocabulário de tamanho reduzido pode ser visto na nuvem de palavras a seguir na figura 6.

3.3. Treinamento do modelo

Após o resumo do modelo, podemos ter uma ideia do número total de pares de entrada e saída de treinamento e validação (desenvolvimento).

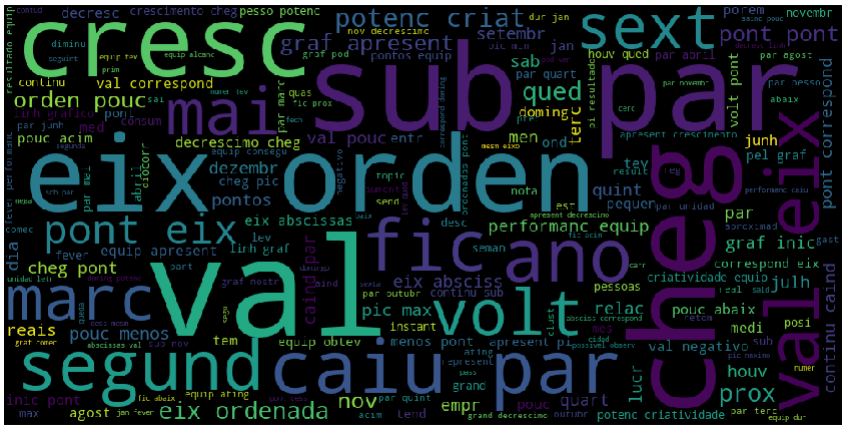

Figura 6. Nuvem de palavras dos itens das legendas da base de dados

realizamos um treinamento com 100 épocas com 7794 amostras e uma validação com 2015 amostras, tendo em média cada época um tempo aproximado de 85s. Após 7693s o melhor resultado de validação foi o seguinte:

- perda no conjunto de dados de treinamento: 2.919
- perda no conjunto de dados de validação: 4.521

3.4. Geração de legenda

Depois de ter modelado e treinado o modelo de rede podemos testar nosso algoritmo e verificar seus resultados tanto em relação aos valores de BLEU quanto em relação empírica.

Realizamos diferentes pesos, sendo os pesos uma tupla, em que cada índice se refere à ordem dos *gram*. Para calcular a pontuação BLEU apenas para correspondências de 1-gram, pode-se especificar um peso de 1 para 1 grama e 0 para 2, 3 e 4 (1, 0, 0, 0). A seguir será indicado os pesos e seus respectivos resultados:

- (1.0, 0, 0, 0) - 0.097744
- (0, 1.0, 0, 0) - 0.018352
- (0, 0, 1.0, 0) - 0.004561
- (0, 0, 0, 1.0) - 0.001778
- (0.5, 0.5, 0, 0) - 0.042353
- (0.3, 0.3, 0.3, 0) - 0.029776
- (0.25, 0.25, 0.25, 0.25) - 0.010983

Realizamos a extração de características da imagem 7, que será utilizada como entrada para a rede, e assim pudemos obter o resultado visto no *frame* a seguir:

startseq graf demostr 5 tip difer pont utiliz mesm reforc reforc questao aco reforc questao aco receb agor menos menos menos menos menos 500 pontos receb de recompens endseq

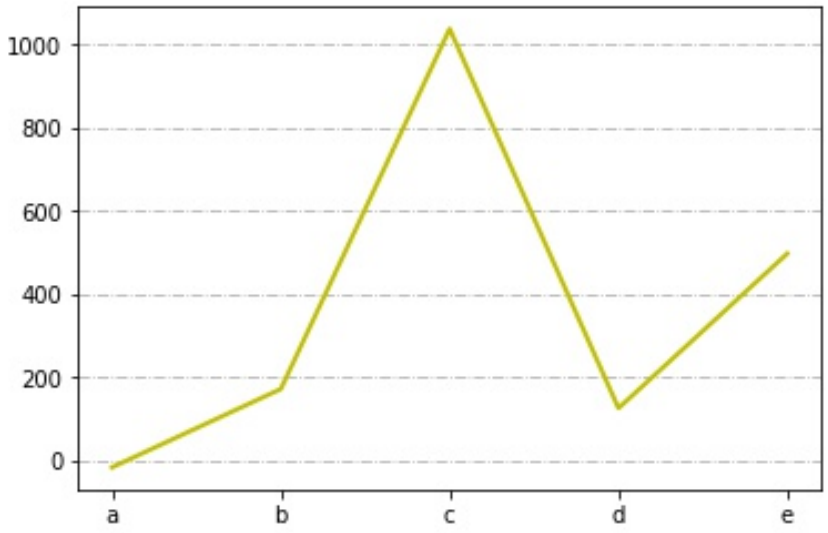

Figura 7. Gráfico utilizado como entrada da rede para teste

4. Conclusões

Nesta pesquisa, estudamos um método de projetar um descritor de imagens baseada em gráficos de linha, realizando extração das características utilizando o modelo pré-treinado de aprendizagem profunda, VGG19. Para que pudéssemos verificar os resultados foram utilizados os métodos de avaliação a partir do BLEU e métodos empíricos de avaliação para assim chegar a uma conclusão final do modelo em si.

O resultado final do estudo revela que o método, mesmo com uma base de dados pequena, nos trouxe resultados interessantes e que devem ser estudados. É possível que o resultado melhore obtendo uma base maior com descrições um pouco melhores, que não tenha tanto *ruído* desnecessário para descrição do gráfico. É interessante estudar a parametrização da rede e outros modelos de extração de características para investigar se há uma melhora nos resultados. Além de outras métricas de avaliação

Referências

[CABRAL et al. 2018] CABRAL, L., LINS, R., and MELLO, R. (2018). *Sumarização Automática Textual Independente de Idioma: Uma plataforma para sumarização textual automática independente de idioma*. Novas Edições Acadêmicas.

[Chollet et al. 2015] Chollet, F. et al. (2015). Keras. `https://keras.io`.

[Hunter 2007] Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95.

[Jason Brownlee 2017] Jason Brownlee (2017). Caption generation with the inject and merge encoder-decoder models. `https://machinelearningmastery.com/caption-generation-inject-merge-architectures-encoder-decoder-model/`. [acessado em 30 de Dezembro de 2019].

[Simonyan and Zisserman 2014] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition.

[Tanti et al. 2017] Tanti, M., Gatt, A., and Camilleri, K. P. (2017). Where to put the image in an image caption generator. *CoRR*, abs/1703.09137.