Plano de Trabalho - Projeto Final

PF06 - Correção ortográfica usando BERT ou T5 - português e inglês
Nome: Rafael Ito (R.A.: 118430)

Título:

NSGC - Neural Spell & Grammar Checker (en/pt)

Sumário:

O uso de corretores ortográficos e gramaticais são muito úteis ao se redigir qualquer tipo de texto. Neste projeto será implementado um corretor ortográfico e gramatical baseado em redes neurais nos idiomas inglês e português utilizando a linguagem de programação Python. Para o corretor de palavras e sentenças em inglês serão explorados os modelos BERT e T5, ambos implementados na biblioteca *transformers*. Para o idioma português apenas o BERT (também disponível no *transformers*) será testado. Espera-se que usando a técnica proposta neste projeto seja possível identificar palavras e sentenças com erros ortográficos ou de gramática e enviar sugestões das respectivas palavras ou frases corretas de volta para o usuário. A métrica usada para ranquear as sugestões é a distância de edição (*edit distance*), calculada entre a suposta palavra errada e a saída dos modelos BERT e/ou T5 dada a frase de contexto com a palavra em análise mascarada. Uma possível aplicação desta técnica poderia ser o uso em buscadores, por exemplo.

Objetivos:

O objetivo deste projeto é elaborar um método para correção ortográfica e gramatical baseado em redes neurais que possa fornecer ao usuário sugestões de correção de palavras e sentenças, usando os modelos BERT e T5 nos idiomas português e inglês. Diversas abordagens e variantes dos modelos serão testadas, buscando-se encontrar o modelo e técnica de melhor custo-benefício. Espera-se obter um algoritmo que seja rápido e confiável que possa ser implementado em contextos onde haja digitação de texto por parte do usuário, como por exemplo em barras de pesquisa e buscadores.

Análise Bibliográfica:

Os principais modelos abordados aqui foram cobertos já em sala de aula, sendo eles o BERT [1] (Devlin et al., 2018) e T5 [2] (Raffel et al., 2019). Para o idioma inglês, ambos modelos e suas variantes com mais ou menos números de parâmetros estão nativamente incluídas na biblioteca *transformers* da *huggingface*. Para o idioma em português há duas versões do BERT na biblioteca da *huggingface*, bert-base-cased e bert-large-cased, ambos

fornecidos pela NeuralMind e com publicação aberta no arXiv, BERT pré-treinado em português [3] (Souza et al., 2019).

Além da sugestão de correção baseada na métrica de distância de edição (*edit distance*), uma arquitetura que aborda o conceito de *embeddings* de sentenças denominada SBERT [4] (Reimers & Gurevych, 2019) será avaliada.

Por fim, o algoritmo será avaliado em dois *datasets*: CoNLL-2014 [5] (Ng et al., 2014) e JFLEG [6] (Napoles et al., 2017).

Datasets a serem utilizados:

Para o idioma inglês, serão usados dois datasets:

- CoNLL-2014 *Shared Task: Grammatical Error Correction* [5]
  https://www.comp.nus.edu.sg/~nlp/conll14st.html

- JFLEG: *A Fluency Corpus and Benchmark for Grammatical Error Correction* [6]
  https://www.aclweb.org/anthology/E17-2037/

Para o idioma português, nenhum *dataset* foi encontrado.

Metodologia:

Inicialmente, utilizaremos o *Google Colab* para fazer os primeiros testes e familiarização com modelos, *datasets* e métricas para ranqueamento de sugestões. Após ter a prova de conceito validada, será analisada a melhor forma de disponibilização para esta ferramenta, podendo ser através de um pacote Python a ser instalado com pip ou de alguma outra maneira.

A primeira estratégia a ser pensada deve ser como ativar o algoritmo que irá disponibilizar as sugestões de correção. Para o caso de palavras digitadas erradas é relativamente simples:

1. O primeiro passo é identificar uma palavra que não se encontra em um vocabulário pré-estabelecido, de preferência que contenha todas ou grande parte das palavras daquele idioma.
2. Em seguida, mascará-se essa palavra em análise e a sequência de contexto é fornecida como entrada para o modelo que pode ser o BERT ou T5 a fim de se prever a palavra mascarada.
3. A partir de uma lista com os *tokens* de maior probabilidade de estar na posição da palavra mascarada, calcula-se alguma métrica, como por exemplo *edit distance*, para avaliar a(s) palavra(s) mais próxima(s) para ser retornada para o usuário como sugestão de correção.

Para o caso de sentenças, pode ser mais complicado. Uma estratégia poderia ser fazer o procedimento anterior para todos os *tokens*, dado que a escrita de todo o texto já foi finalizada. Entretanto, isso pode ser computacionalmente custoso a medida que o texto aumenta de tamanho. Para buscadores e barras de pesquisa isso não deve ser um problema, visto que as sequências para este fim costumam ser curtas. Para outros tipos de texto, alguma estratégia deverá ser analisada.

Todo código fonte pode ser encontrado neste repositório do GitHub e o *notebook* com o desenvolvimento pode ser encontrado neste link para o Colab.

Cronograma:

Considerando o período entre os dias 21/05 e 25/06, temos exatamente 5 semanas para trabalhar com o projeto. Por mais que possa parecer um período suficiente, ele pode ficar curto caso não haja um planejamento eficiente de trabalho.

- Semana 1: 21/maio → 27/maio (estudos iniciais)
    - Estudos direcionados.
    - Leitura dos artigos envolvidos.
    - Familiarização com *datasets*.
    - Primeira proposta de projeto em detalhes.

- Semana 2: 28/maio → 03/junho (experimentações)
    - Teste de modelos com diferentes números de parâmetros (base, large, etc).
    - Comparação em termos de qualidade e custo computacional de corretores baseados em BERT e T5.
    - Uso de diferentes métricas para calcular palavras/sentenças mais próximas da que está em análise (*edit distance*, SBERT).

- Semana 3: 04/junho → 10/junho (entrega do correto em inglês)
    - Compilação dos diferentes testes produzidos na semana anterior.
    - Proposta da arquitetura final.
    - Finalização do corretor para inglês.

- Semana 4: 11/junho → 17/junho (entrega do corretor em português)
    - Aplicação do procedimento que melhor funcionou para o inglês, mas agora com textos em português.
    - Possíveis pequenos ajustes devido a mudança de idioma.
    - Finalização do corretor para português.

- Semana 5: 18/junho → 24/junho (disponibilização do algoritmo e documentação)
    - Revisão de código.
    - Encapsulação do código final em pacotes.
    - Finalização de documentação e relatório.

- 25/junho → apresentação do projeto

Referências:

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018

[2] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv preprint arXiv:1910.10683, 2019.

[3] Souza, F., Nogueira, R., Lotufo, R., 2019. Portuguese named entity recognition using bert-crf. arXiv preprint arXiv:1909.10649.

[4] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERTnetworks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2019.

[5] Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 shared task on grammatical error correction. In Proceedings of the 18th Conference on Computational Natural Language Learning: Shared Task, pages 1–14.

[6] Napoles, C.; Sakaguchi, K.; and Tetreault, J. 2017. JFLEG: A fluency corpus and benchmark for grammatical error correction. In Proc. EACL.