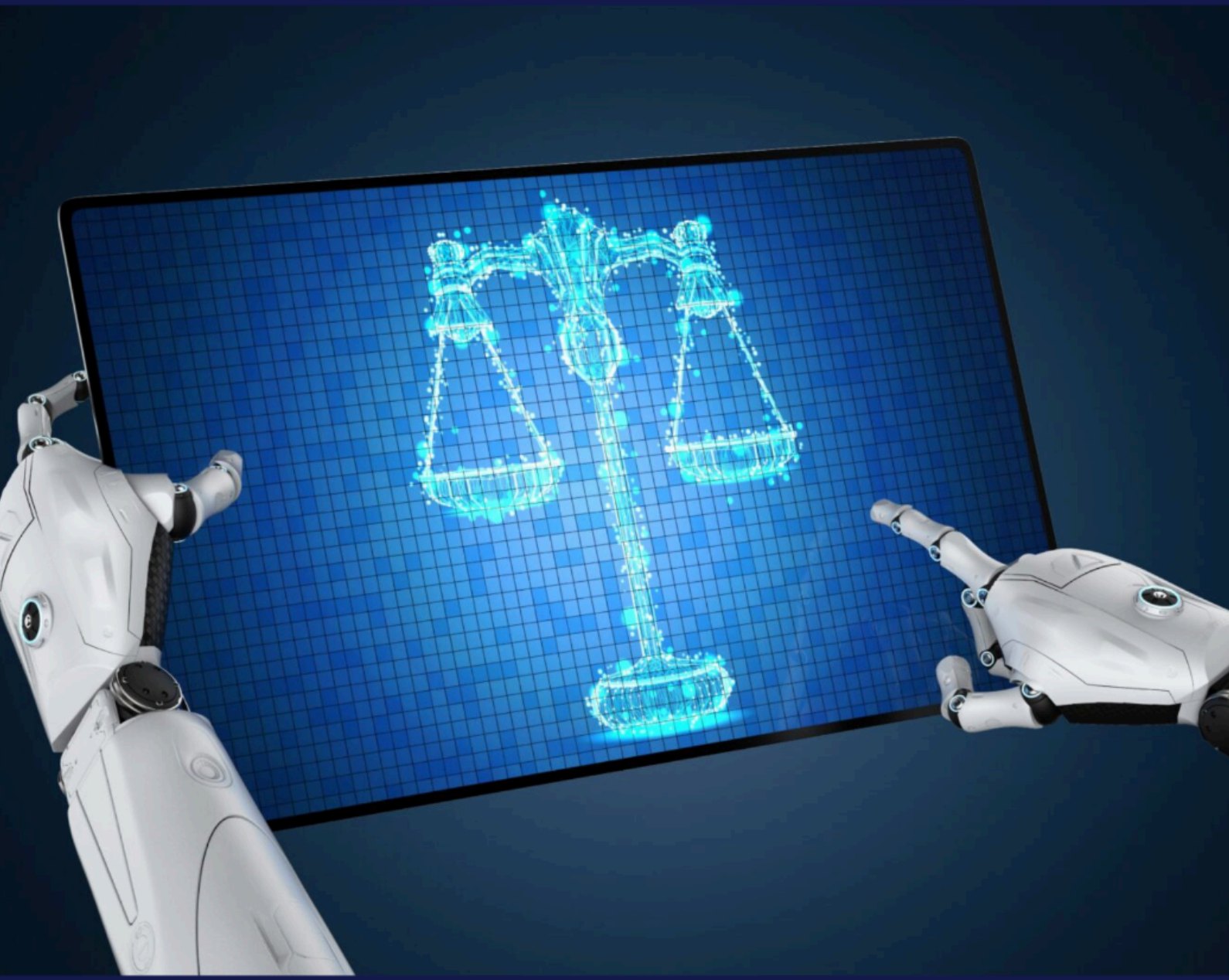
UNIVERSIDADE FEDERAL DE GOIÁS / INSTITUTO DE INFORMÁTICA

Processamento de Linguagem Natural

Aplicado em textos jurídicos

Héber Júnior Rodrigues Amaral



UFG
UNIVERSIDADE
FEDERAL DE GOIÁS

UNIVERSIDADE FEDERAL DE GOIÁS (UFG)
INSTITUTO DE INFORMÁTICA (INF)

HÉBER JÚNIOR RODRIGUES AMARAL

PROCESSAMENTO DE LINGUAGEM NATURAL

Aplicado em textos jurídicos

Goiânia
2024

UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TRABALHO DE CONCLUSÃO DE CURSO DE GRADUAÇÃO NO REPOSITÓRIO INSTITUCIONAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio do Repositório Institucional (RI/UFG), regulamentado pela Resolução CEPEC no 1240/2014, sem ressarcimento dos direitos autorais, de acordo com a Lei no 9.610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou download, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo dos Trabalhos de Conclusão dos Cursos de Graduação disponibilizado no RI/UFG é de responsabilidade exclusiva dos autores. Ao encaminhar(em) o produto final, o(s) autor(a)(es)(as) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do Trabalho de Conclusão de Curso de Graduação (TCCG)

Nome(s) completo(s) do(a)(s) autor(a)(es)(as): **HEBER JUNIOR RODRIGUES AMARAL**

Título do trabalho:

PROCESSAMENTO DE LINGUAGEM NATURAL

Aplicado em textos jurídicos

2. Informações de acesso ao documento (este campo deve ser preenchido pelo orientador) Concorda com a liberação total do documento [**X**] SIM [ ] NÃO[1]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante: a) consulta ao(à)(s) autor(a)(es)(as) e ao(à) orientador(a); b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo do TCCG. O documento não será disponibilizado durante o período de embargo.

Casos de embargo:

- Solicitação de registro de patente;

- Submissão de artigo em revista científica;

- Publicação como capítulo de livro.

Obs.: Este termo deve ser assinado no SEI pelo orientador e pelo autor.

Documento assinado eletronicamente por **Héber Júnior Rodrigues Amaral**, **Discente**, em 16/02/2024, às 09:50, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Fernando Marques Federson**, **Professor do Magistério Superior**, em 12/09/2024, às 11:03, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **4383380** e o código CRC **9DB2A2A2**.

Referência: Processo nº 23070.008383/2024-11 SEI nº 4383380

HÉBER JÚNIOR RODRIGUES AMARAL

PROCESSAMENTO DE LINGUAGEM NATURAL

Aplicado em textos jurídicos

Relatório final de Trabalho de Conclusão de Curso, apresentado à Universidade Federal de Goiás, como parte das exigências para a obtenção do título de Bacharel em Inteligência Artificial.
Orientador: Prof. Dr. Fernando Marques Federson

Goiânia
2024

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

AMARAL, HEBER JUNIOR RODRIGUES
PROCESSAMENTO DE LINGUAGEM NATURAL [manuscrito] : Aplicado em textos jurídicos / HEBER JUNIOR RODRIGUES AMARAL. - 2024.
76 f.

Orientador: Prof. Dr. FERNANDO MARQUES FEDERSON.
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Goiás, Instituto de Informática (INF), Inteligência Artificial, Goiânia, 2024.

1. inteligência artificial. 2. processamento de linguagem natural. 3. textos jurídicos. I. FEDERSON, FERNANDO MARQUES, orient. II. Título.

CDU 004

HÉBER JÚNIOR RODRIGUES AMARAL

PROCESSAMENTO DE LINGUAGEM NATURAL

Aplicado em textos jurídicos

Relatório final de Trabalho de Conclusão de Curso, apresentado à Universidade Federal de Goiás, como parte das exigências para a obtenção do título de Bacharel em Inteligência Artificial.

Data da Aprovação: 08 de fevereiro de 2024.

Prof. Dr. Fernando Marques Federson
Orientador (INF-UFG)

Prof. Dr. Aldo André Díaz Salazar
Coordenador de TCC do BIA (INF-UFG)

Prof. Dr. Vinícius Sebba Patto
Coordenador do BIA (INF-UFG)

Prof. Dr. Iwens Gervasio Sene Junior
(INF-UFG)

HÉBER JÚNIOR RODRIGUES AMARAL

PROCESSAMENTO DE LINGUAGEM NATURAL

Aplicado em textos jurídicos

RESUMO

Este Relatório de Conclusão de Curso tem como objetivo reunir os resultados da minha jornada para me tornar um especialista em **Processamento de Linguagem Natural (texto jurídico)**. Uma ilustração e sua narrativa descrevem os períodos de trabalho. Os Apêndices contêm os Termos de Aceite de Entrega e os resultados obtidos durante cada período de trabalho.

Palavras-chave: inteligência artificial, processamento de linguagem natural, textos jurídicos.

ABSTRACT

This Course Completion Report aims to bring together the results of my journey to become an expert in **Natural Language Processing (legal text)**. An illustration and its narrative describe the work periods. The Appendices contain the Delivery Acceptance Terms and the results obtained during each work period.

Keywords: artificial intelligence, natural language processing, legal texts.

Goiânia

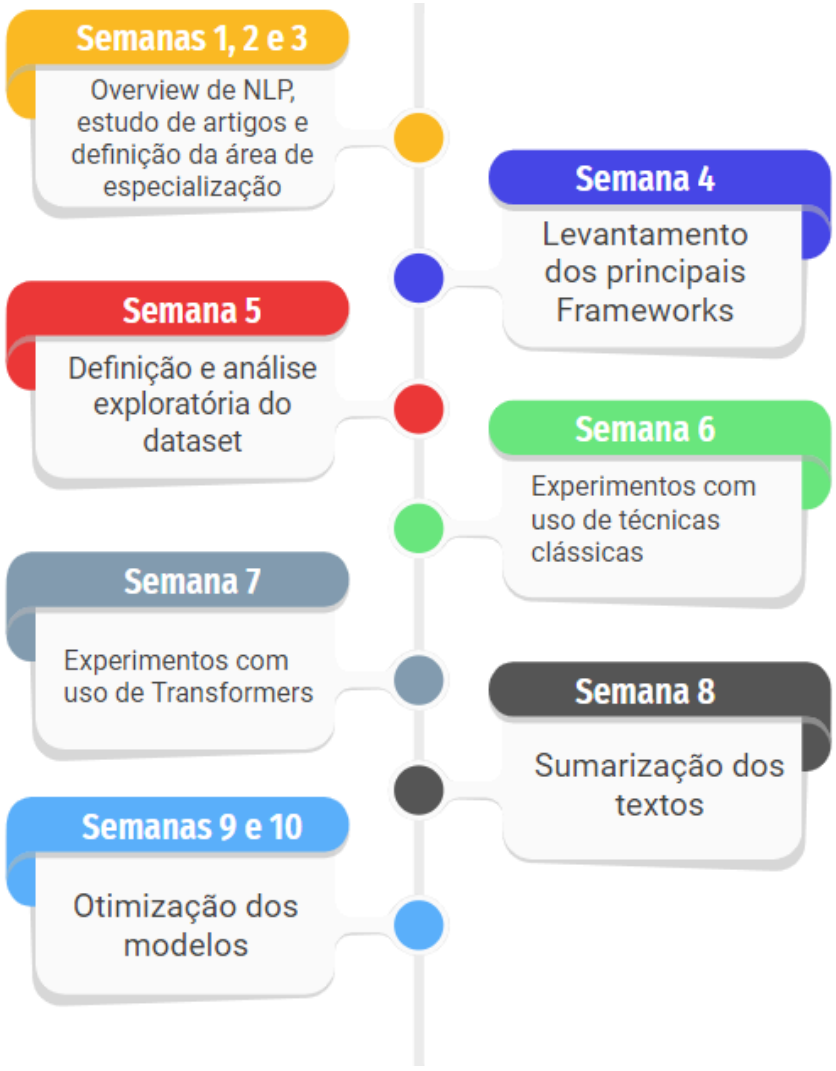
2024

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

Minha Jornada

Héber Júnior R. Amaral

Especialista em: Processamento de Linguagem Natural (texto jurídico)



Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

MINHA JORNADA

Nome: Héber Júnior Rodrigues Amaral

Especialidade: Processamento de Linguagem Natural (texto jurídico)

Objetivo deste documento

Durante o processo da disciplina Residência em IA[1], foram gerados diversos resultados na construção da minha especialização. A cada semana, um conjunto de resultados foi formalizado por um Termo de Aceite de Entrega e avaliado por uma banca, considerando o planejado e o realizado para o período. Este documento tem como objetivo descrever esses resultados obtidos, fazendo referência aos Termos de Aceite de Entrega e seus documentos associados.

Minha Jornada

Na primeira **Semana**, com base na *Conference on Computational Science and Computational Intelligence* (CSCI'23) e em experiências vividas durante o Bacharelado em Inteligência Artificial, defini a área de conhecimento que seria o foco da minha especialização. Optei por trabalhar com Processamento de Linguagem Natural (NLP). Essa é uma área muito ampla e com diversas aplicações possíveis, mas uma em particular que chama muito minha atenção é NLP aplicado ao domínio jurídico. Por isso, fiz a leitura do artigo sobre NLP no âmbito jurídico de Katz[2], que faz uma revisão abrangente do estado atual do NLP no domínio jurídico e destaca as principais tendências e desafios enfrentados pelos pesquisadores. O artigo também cita algumas tarefas que podem ser trabalhadas nessa área, como sumarização, classificação e extração de informações. Assim, pude perceber que é uma área que vale a pena ser explorada e poderia agregar muito durante minha formação. Mais detalhes da primeira **Semana** podem ser encontrados no **Apêndice 1**.

Na **Semana 2**, foi feita uma revisão bibliográfica simples com o estudo de *surveys* de NLP, em especial, artigos que tratavam da tarefa de classificação de textos, uma das tarefas

[1] Dez semanas, entre setembro de 2023 e janeiro de 2024.
[2] Katz, D. M. et al. *Natural Language Processing in the Legal Domain*; 2023.

que foram abordadas na minha jornada. Além disso, elaborei um documento com os principais termos quando falamos de NLP. Esse documento cita as principais formas de representação de palavras, discute o uso de métodos clássicos e também o uso de métodos baseados em deep learning para a classificação de textos e NLP de forma geral. Todos os artigos estudados e o documento que foi elaborado podem ser encontrados no **Apêndice 1**.

A partir da definição da área e revisão dos principais conceitos, durante a **Semana 3**, aprofundei no estudo da tarefa de classificação de textos no domínio jurídico. Pesquisei sobre como o problema de classificação estava sendo abordado no contexto escolhido, quais técnicas eram usadas e quais datasets disponíveis. Fiz a leitura do artigo *"Text Classification in Juridical Field"* que destaca diferentes formas de representações de textos e mostra que em determinados contextos, o uso de técnicas clássicas podem superar as técnicas chamadas "estado da arte". Além disso, fiz a leitura do artigo sobre o conjunto de dados (dataset) VICTOR[3], criado feito pela Universidade de Brasília (UnB) em parceria com o Supremo Tribunal Federal (STF). Com base nesse artigo, também pude definir que esse conjunto de dados seria utilizado na minha jornada. Todos os detalhes das atividades dessa **Semana** também estão no **Apêndice 1**.

Após construir a base teórica nas primeiras três **Semanas**, na quarta **Semana** comecei trabalhar com a parte "prática" da minha jornada. Realizei uma pesquisa dos principais frameworks usados em NLP. Essa pesquisa foi feita pensando principalmente na tarefa de classificação de textos e também nas etapas desde a importação dos dados até a otimização de hiperparâmetros do modelo. A pesquisa feita destacou a importância de etapas como importação de dados, pré-processamento, extração de *features*, treinamento de modelos, obtenção de métricas e otimização de hiperparâmetros. Listei diversos frameworks e descrevi suas características e aplicações específicas no contexto de NLP. Este estudo está no **Apêndice 2**.

Na **Semana 5**, foi quando aprofundei a minha experimentação. Nessa **Semana**, realizei uma análise exploratória do dataset VICTOR, usado na tarefa de classificação de textos jurídicos. Na análise, destaquei pontos como a distribuição das classes onde pude perceber o grande desbalanceamento do dataset, grande parte das sentenças estavam entre 256 e 512 *tokens*, ou seja, são sentenças que não são curtas e o contexto pode ser

[3] Araujo, P. H. L. et al. *VICTOR: a dataset for Brazilian legal documents classification*; 2020.

um fator muito importante a se considerar. Além disso, observei as palavras mais comuns em cada classe e elas eram bem semelhantes entre as classes, o que leva a acreditar que modelos que lidam com contexto são mais indicados para essa tarefa. A análise exploratória detalhada se encontra no **Apêndice 3**.

Já na **Semana 6**, comecei a realização de experimentos de classificação de textos com o dataset VICTOR. O *pipeline* seguido foi o seguinte: importação dos dados, realização do pré-processamento nos textos, remoção de *stopwords*, *lowercase*, remoção de caracteres especiais, representação dos textos com métodos clássicos, uso de algoritmos classificadores clássicos (como *Decision Tree*) e avaliação dos resultados com uso da métrica F1-score. Nesse caso, foi utilizada a representação de palavras conhecida como *Term Frequency-Inverse Document Frequency* (TF-IDF). Além disso, foi atingido o valor de 0,78 no F1-score. O que mostra que, mesmo o TF-IDF sendo uma abordagem mais simples, pode ser eficaz para destacar palavras únicas e informativas para cada classe, mesmo que as mais comuns sejam semelhantes. Os detalhes dos experimentos estão no **Apêndice 4**.

Na **Semana 7**, pensando em melhorar o F1-score e considerando que o contexto é muito importante em textos jurídicos, segui o *pipeline* da tarefa de classificação de textos, mas agora utilizando abordagens baseadas na arquitetura Transformers. Nesta semana, realizei o *fine-tuning* de modelos pré-treinados, como o BERT e RoBERTa. De forma geral, pode-se perceber uma melhora significativa ao utilizar modelos baseados na arquitetura Transformers, com o valor de F1-score chegando a 0,87. Isso pode ser fruto de alguns fatores, entre eles: a capacidade de lidar com sequências longas, a transferência de aprendizado e a capacidade de capturar relações semânticas complexas e contextualizadas em todo o texto. Apesar do ganho no valor de F1-score, também vale destacar que o uso de modelos baseados na arquitetura Transformers demanda mais tempo de treinamento e mais recursos computacionais. Todos os detalhes podem ser encontrados no **Apêndice 5**.

Na **Semana 8**, trabalhei com a sumarização dos textos, pensando em abordar outra tarefa quando falamos de NLP no domínio jurídico e também pensando em melhorar o *pipeline* de classificação de textos. A ideia foi sumarizar os textos onde o número de *tokens* fosse maior que 512. Essa estratégia trouxe resultados positivos melhorando o F1-score do modelo (na tarefa de classificação de textos). Indicando que muitas vezes, depois do limite de 512 *tokens*, tínhamos a perda de informações importantes e que, quando realizada a

sumarização dos textos longos, essas informações permaneceram no texto e melhoraram o desempenho do modelo. Os detalhes dessa atividade podem ser encontrados no **Apêndice 6**.

Durante as **Semanas 9 e 10**, fiz a otimização dos modelos de classificação de textos visando reduzir o tempo de inferência e possibilitando que eles pudessem ser executados em hardwares com menos recursos computacionais (sem a necessidade de GPUs). Na nona **Semana**, trabalhei com a quantização. Essa é uma técnica de redução do número de bits necessários para representar os parâmetros de um modelo que visa reduzir o tamanho e o tempo de inferência, mas com perda mínima de desempenho. Com o uso dessa técnica foi possível obter bons resultados como um modelo menor, tempo de inferência reduzido e uma perda pequena de desempenho. Já na décima **Semana**, trabalhei com modelos do tipo distil. Nesse caso, um modelo mais compacto, chamado de "aluno", é treinado para reproduzir o comportamento de um modelo maior e mais complexo, denominado "professor" ou "modelo de ensino". Essa abordagem também trouxe bons resultados, diminuindo o tempo de inferência que era o objetivo da otimização. Os detalhes das atividades de otimização estão no **Apêndice 7**.

Após as dez semanas vividas na minha jornada, acredito que foi possível construir uma boa base para me tornar um especialista em Processamento de Linguagem Natural, em especial, aplicado em textos jurídicos. As fases teóricas e práticas trouxeram conhecimentos importantes, desde a identificação de desafios inerentes ao contexto jurídico até a implementação de abordagens práticas como o uso de Transformers. A análise exploratória, os experimentos e as otimizações dos modelos proporcionaram uma visão abrangente, ressaltando não apenas a eficácia de diferentes técnicas, mas também a necessidade de equilibrar desempenho do modelo e eficiência computacional. Essa jornada não apenas marcou um ponto crucial em minha formação, mas também estabeleceu um sólido alicerce para futuras explorações e contribuições no campo do NLP. Pretendo continuar aprofundando mais ainda nessa área abordando novas tarefas (como *Questions Answers* e Reconhecimento de Entidades Nomeadas) e novos métodos, como o uso dos *Large Language Models* (LLMs) e acredito que tudo que foi estudado servirá de base para os próximos passos.

APÊNDICE 1

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 19 de out. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Héber Júnior, Pedro Koziel

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Área de especialização: Processamento de Linguagem Natural com foco em classificação de textos

O sistema judiciário brasileiro é sobrecarregado, com milhões de processos aguardando julgamento, e a espera por uma decisão pode se estender por vários anos. Os recursos e o tempo gastos são enormes, impactando a eficiência do sistema, a segurança jurídica e colocando uma carga significativa sobre os orçamentos públicos.

Por isso, esse projeto tem o objetivo de automatizar o processo de **classificação de textos jurídicos** usando técnicas de **Processamento de Linguagem Natural (NLP) e Machine Learning (ML)**. Ao automatizar essa tarefa, não apenas economizamos recursos financeiros, mas também permitimos que profissionais do direito direcionem sua energia para tarefas que exigem julgamento, discernimento e experiência humana. Isso não só aumenta a eficiência do sistema judiciário, mas também contribui para uma redução significativa da sobrecarga que afeta esse sistema.

Durante a realização desse projeto, serão abordadas técnicas clássicas para representação dos textos como **TF–IDF** (term frequency–inverse document frequency) com o uso de algoritmos classificadores como regressão logística e também serão utilizadas técnicas consideradas estado da arte como as baseadas na **arquitetura transformers** (Bert e Roberta, por exemplo). Considerando que textos jurídicos são longos e o contexto muitas vezes é essencial, o uso de representações transformers se faz muito necessário e essa técnica tem se mostrado muito eficiente na

representação de palavras/textos. Outras etapas também são muito importantes e vão fazer parte desse projeto: análise de dados, pré–processamento dos textos, escolha de métricas para avaliação. Além disso, também serão exploradas técnicas de otimização de hiperparâmetros e técnicas de otimização/quantização de modelos.

Pensando nisso, estou usando alguns artigos como referência:

O primeiro deles (Natural Language Processing in the Legal Domain - https://arxiv.org/pdf/2302.12039.pdf) trata sobre o uso de NLP no ambiente jurídico de uma forma mais geral. Esse artigo aborda diferentes tarefas, entre elas, a tarefa de classificação de textos jurídicos.

Outros artigos interessantes são os que tratam sobre o estado da arte que temos quando falamos em NLP, em especial a tarefa de classificação. Hoje utilizamos modelos baseados na arquitetura transformers e abaixo listo alguns artigos que vão apoiar essa pesquisa:

Transformers: https://arxiv.org/pdf/1706.03762.pdf , Bert: https://arxiv.org/pdf/1810.04805.pdf , Roberta: https://arxiv.org/pdf/1907.11692.pdf

Áreas da Conference on Computational Science and Computational Intelligence (CSCI'23) que têm relação com o projeto proposto: Processamento de Linguagem Natural; Resolução automatizada de problemas; Redes Neurais e Aplicações; Aprendizagem Supervisionada.

Planejamento: **[descrever o que pretende fazer para realizar a próxima ENTREGA]**

- Realizar uma Análise Exploratória de Dados (EDA) para entender a distribuição das classes e características dos textos.

- Identificar quaisquer desequilíbrios de classe e considerar estratégias para lidar com eles.

- Visualizar dados e estatísticas descritivas para obter insights.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: **26 de out. de 2023**

Participantes da Entrega **[matriculados em Residência em IA]:**

Héber Júnior, Pedro Koziel e Pedro Augusto

Entrega: **[descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]**

Durante essa semana e após os encontros da semana passada, compreendemos melhor o que se espera de nós dessa residência. Assim, realizamos a busca por surveys de NLP, tanto no aspecto geral quanto específico da parte de classificação, a fim de ter um histórico de técnicas e métodos da área, e também buscar quais dessas técnicas e métodos são os que possuem melhores resultados atualmente. Os surveys podem ser encontrado em:

- A Survey on Text Classification: From Traditional to Deep Learning (https://arxiv.org/pdf/2008.00364.pdf)

- A Survey on Text Classification Algorithms: From Text to Predictions (https://www.mdpi.com/2078-2489/13/2/83)

- Efficient Methods for Natural Language Processing: A Survey (https://arxiv.org/pdf/2209.00099.pdf)

Também foi montado um repositório compartilhado de artigos entre todos da turma que estão trabalhando com NLP e/ou LLM:
https://drive.google.com/drive/folders/11igIGGlTPdB_qAXi71--XOu71Rxv3cWH

Junto com esse estudo, também fizemos um compilado de termos que julgamos ser mais importante para essa etapa inicial da residência, que constitui os fundamentos de NLP. Para isso foi utilizado de base tanto os surveys acima quanto os artigos que foram mencionados na entrega da semana passada. Esse compilado pode ser acessado em: gate 26/10/2023

Planejamento: **[descrever o que pretende fazer para realizar a próxima ENTREGA]**

Como na entrega dessa semana o foco foi aprofundar na classificação de textos de forma geral, para a próxima entrega o objetivo é aprofundar na classificação de **textos jurídicos**. Para isso, o planejamento é:

- Estudo de artigos sobre classificação de textos especificamente no **domínio jurídico**
  - **Text Classification and Prediction in the Legal Domain** (https://aclanthology.org/2022.lrec-1.504.pdf)
  - **Text Classification in the Brazilian Legal Domain** (https://www-di.inf.puc-rio.br/~casanova/Publications/Papers/2022-Papers/2022-ICEIS-Coelho.pdf)
- Buscar **datasets** usados para classificação de textos jurídicos

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

[Documento citado no Termo de Aceite de Entrega de 26 de outubro]

Revisão de conceitos

Usando os surveys e artigos que foram citados como referência nos termos de entrega da semana passada e essa, separamos os principais tópicos sobre representações de palavras e modelos de NLP.

1. Representação de Palavras: A representação de palavras é fundamental para a classificação de textos, pois transforma o texto em um formato que os modelos de aprendizado de máquina podem entender e, consequentemente, possibilitar seu processamento e realização de diversas tasks, como classificação e sumarização. Existem várias abordagens, incluindo:

a. Bag of Words (BoW): Nessa abordagem, um texto é representado como um conjunto de palavras, ignorando a ordem das palavras. Cada palavra é codificada como um vetor de recursos. As frequências das palavras são contadas e usadas como valores nos vetores.

b. TF-IDF (Term Frequency-Inverse Document Frequency): O TF-IDF atribui valores a palavras com base em sua frequência em um documento específico e sua importância geral em todo o corpus de documentos, o que ajuda a destacar palavras importantes.

c. Word Embeddings Estáticos (word2vec): Além das abordagens tradicionais, é importante mencionar o uso de word embeddings estáticos, como word2vec. Esses embeddings representam palavras como vetores em um espaço de alta dimensão, capturando relações semânticas e contextualmente relevantes entre palavras. O word2vec é treinado em

grandes conjuntos de texto e é uma maneira eficaz de capturar o significado das palavras para uso em tarefas de processamento de linguagem natural.

2. Classificadores Clássicos: Os classificadores clássicos podem ser usados com representações de palavras como BoW ou TF-IDF. Alguns exemplos de classificadores clássicos incluem:

a. Naive Bayes: É um classificador probabilístico que assume independência entre as palavras.

b. Regressão Logística: Uma técnica que modela a probabilidade de uma instância pertencer a uma classe em função das variáveis explicativas (representação de palavras).

c. SVM (Support Vector Machine): mapeia os vetores de palavras para um espaço de alta dimensão e encontra hiperplanos de separação entre classes.

3. Métodos Baseados em Deep Learning: Os métodos baseados em deep learning têm se destacado na classificação de textos, especialmente quando combinados com representações de palavras mais avançadas. Alguns métodos incluem:

a. Redes Neurais Artificiais (MLP): Redes neurais de feedforward, como perceptrons multicamadas (MLP), podem ser usadas para classificação de textos. Eles podem ser alimentados com representações vetoriais de palavras.

b. Redes Neurais Recorrentes (RNN): As RNNs são capazes de lidar com sequências de palavras e podem capturar dependências temporais. No entanto, elas têm dificuldade em lidar com sequências muito longas e sofrem de desvanecimento de gradientes (vanishing gradients).

c. Redes Neurais com Mecanismo de Atenção (Attention): As redes com atenção, como o mecanismo de atenção de seq2seq, são capazes de dar mais peso a certas partes do texto durante a classificação, o que é útil para entender o contexto.

d. Transformers: Os modelos baseados em Transformers revolucionaram o campo do processamento de linguagem natural (NLP) e a classificação de textos. Eles introduziram uma arquitetura de rede neural altamente paralelizável que se destaca em uma variedade de tarefas NLP, incluindo classificação de textos.

4. Arquitetura dos Transformers: A principal inovação dos Transformers é a arquitetura de atenção, que permite que o modelo considere todas as palavras do contexto ao mesmo tempo. Isso é fundamental para a compreensão do contexto e a captura de relações de longo alcance. Cada camada do Transformer contém duas partes principais:

a. Mecanismo de Auto-Atenção (Self-Attention): É a espinha dorsal do Transformer. Ele permite que o modelo avalie a importância de todas as palavras na sequência em relação a uma palavra de entrada específica. Isso é feito por meio de pesos de atenção que indicam a importância relativa de cada palavra para a palavra de referência.

b. Redes de Feedforward (Feedforward Networks): Após a auto-atenção, uma camada de redes neurais de feedforward é aplicada para combinar informações e gerar saídas finais.

Os modelos Transformers são altamente adaptáveis para tarefas de classificação de textos. Para usar um modelo Transformer para classificação, a camada de saída pode ser personalizada para se ajustar ao número de classes do problema de classificação. Geralmente, é adicionada uma camada de saída softmax no topo da arquitetura para calcular as probabilidades de pertencer a cada classe.

Uma das vantagens dos modelos Transformers é a capacidade de pré-treinamento em grandes quantidades de dados e, em seguida, ajustar o modelo para tarefas específicas por meio do fine-tuning. Para classificação de textos, é possível pré-treinar um modelo Transformer em um grande corpus de texto e, em seguida, ajustá-lo em dados de treinamento de classificação específicos.

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: **9 de nov. de 2023**

Participantes da Entrega **[matriculados em Residência em IA]**:

Héber Júnior

Entrega: **[descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]**

Na última entrega foi feito o estudo de surveys que tratavam do problema de classificação de textos de modo geral. Nessa entrega, a ideia foi aprofundar mais no estudo da tarefa de classificação, agora focando especificamente no domínio jurídico. Para isso, busquei artigos e datasets que pudessem apoiar esse trabalho.

A ideia era entender como a tarefa de classificação de textos estava sendo abordada no contexto jurídico, quais técnicas estão sendo usadas, quais datasets disponíveis, melhores estratégias e entender as características específicas dos textos jurídicos.

Os principais artigos usados para apoiar o estudo:

Text Classification and Prediction in the Legal Domain – técnicas estado da arte como o BERT

Text Classification in the Brazilian Legal Domain – técnicas tradicionais, Doc2Vec com SVM, SMOTE

VICTOR: a dataset for Brazilian legal documents classification – criação do dataset e abordagem de técnicas clássicas até técnicas avançadas

Text Classification in juridical field – TF-IDF e BERT

Nesse link tem um resumo de cada artigo estudado, descrevendo a metodologia usada neles: Gate 09-11

Planejamento: **[descrever o que pretende fazer para realizar a próxima ENTREGA]**

Levantamento das principais ferramentas/frameworks utilizadas em cada uma das etapas de um trabalho de classificação de textos, considerando desde a obtenção dos dados até a avaliação dos modelos (fazendo também uma comparação, mostrando vantagens e desvantagens de cada uma e

quando utilizar cada ferramenta). Além disso, entender a relação dos fundamentos estudados até então com as ferramentas que serão utilizadas.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

[Documento citado no Termo de Aceite de Entrega de 09 de novembro]

Resumo de artigos estudados

Text Classification and Prediction in the Legal Domain

O primeiro trabalho estudado foi o Text Classification and Prediction in the Legal Domain (https://aclanthology.org/2022.lrec-1.504.pdf). Esse artigo descreve a aplicação de técnicas de processamento de linguagem natural (NLP) em contextos jurídicos, com ênfase na classificação de respostas de companhias aéreas em reclamações de compensação de voos. O foco principal é a classificação das respostas das companhias aéreas em cinco categorias: Liquidação, Negação, Pago Direto, Negociação Direta e Necessidade de Informações Adicionais.

A metodologia emprega modelos baseados na arquitetura transformer, como o BERT, para realizar a classificação de texto. Esses modelos são conhecidos por sua capacidade de capturar informações contextuais em textos longos e complexos, tornando-os adequados para tarefas de classificação de textos no domínio jurídico.

Uma parte interessante abordada na metodologia deste artigo é a integração dos modelos no fluxo de trabalho jurídico existente. Os modelos são usados para classificar automaticamente as respostas das companhias aéreas, fornecendo suporte às equipes jurídicas no processamento de casos.

Text Classification in the Brazilian Legal Domain
(https://www-di.inf.puc-rio.br/~casanova/Publications/Papers/2022-Papers/2022-ICEIS-Coelho.pdf)

Outro artigo estudado foi o Text Classification in the Brazilian Legal Domain. Esse artigo propõe um novo modelo chamado MuDEC (Multi-step Document Embedding-Based Classifier) para classificar pareceres jurídicos relacionados a reclamações de consumidores com base no valor do dano moral. O modelo combina a técnica Doc2vec e SVM para extração e classificação de recursos, respectivamente. O artigo enfatiza a importância de lidar com conjuntos de dados desequilibrados e sugere o uso da técnica conhecida como

SMOTE (gerando exemplos sintéticos para a classe minoritária, a fim de equilibrar as proporções das classes no conjunto de dados).

A metodologia usada neste trabalho usa a extração de recursos com Doc2vec. O Doc2vec é um método aplicado para extrair recursos dos pareceres. Este modelo é treinado exclusivamente nos dados de treinamento e converte cada instância textual em um vetor numérico de dimensão pré-definida. Após isso, um classificador SVM (Support Vector Machine) é empregado para atribuir cada instância a uma das classes possíveis de valor de dano moral.

VICTOR: a dataset for Brazilian legal documents classification
(https://aclanthology.org/2020.lrec-1.181.pdf)

O artigo descreve o conjunto de dados VICTOR, criado a partir de documentos jurídicos do Supremo Tribunal Federal do Brasil. O conjunto de dados oferece suporte a duas tarefas: a classificação de tipos de documentos e a atribuição de temas, que é um problema de classificação multi-label.

O objetivo principal do trabalho é aplicar técnicas de Processamento de Linguagem Natural (PLN) e Aprendizado de Máquina (ML) para analisar casos do Supremo Tribunal Federal do Brasil, visando contribuir para a redução da grande quantidade de ações judiciais e do tempo médio de tramitação. O STF lida com cerca de 42 mil processos a cada semestre, e a digitalização desses processos é um desafio devido à diversidade intraclasse e à qualidade dos documentos digitalizados.

O artigo explora diferentes métodos de classificação de tipos de documentos, incluindo abordagens baseadas em bag of words com os classificadores Naive Bayes e SVM, redes neurais convolucionais (CNN) e redes neurais recorrentes (LSTM).

Outro ponto interessante de destacar é o artigo “Text Classification in juridical field” (https://medium.com/@saadtazroute_84430/text-classification-in-juridical-field-da2d1c0a927c) que compara diferentes técnicas de classificação e mostra que dependendo do contexto, os métodos clássicos e mais simples podem apresentar resultados superiores a modelos considerados estado da arte.

APÊNDICE 2

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: **16 de nov. de 2023**

Participantes da Entrega **[matriculados em Residência em IA]:**

Héber Júnior, Pedro Augusto e Pedro Koziel

Entrega: **[descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]**

Cada integrante do grupo vai para uma sub-área diferente, porém todas estão dentro da task de classificação de texto. Assim, o geral da atividade realizada foi igual para todos os integrantes, mas alguns detalhes foram específicos para cada um.

Como mencionado no planejamento anterior, durante essa semana desenvolvemos um levantamento com os principais frameworks para realização da task de classificação de textos. O objetivo desse levantamento é que, como especialistas, ao se deparar com algum problema dessa área de NLP já sejamos capazes de, antes mesmo de começar a parte de programação, já identificar quais frameworks são mais adequados ao problema em questão. Esse levantamento pode ser encontrado em: Levantamento de Frameworks

Embora tenhamos realizado as atividades desta semana em conjunto, o escopo que estou abordando durante a residência é a tarefa de classificação de textos jurídicos. Realizamos esse estudo em grupo, visto que, quando chegar a hora de realizar a aplicação prática, todos possamos recorrer a esse levantamento para verificar as ferramentas que melhor se encaixam em cada área de aplicação específica.

Planejamento: **[descrever o que pretende fazer para realizar a próxima ENTREGA]**

Para a próxima entrega, o objetivo é realizar uma **análise exploratória do dataset** VICTOR (para entender a distribuição de cada uma das classes, características dos textos, etc), que é um dataset criado pelo laboratório de Inteligência Artificial da UnB em parceria com o Supremo Tribunal Federal (STF). Esse é um dataset criado para ser usado na classificação de textos jurídicos, com o objetivo de auxiliar pesquisadores dessa área e também para que os pesquisadores possam mostrar suas descobertas e melhorias ao utilizar esse conjunto de dados, colaborando assim para o desenvolvimento da pesquisa acadêmica. Além disso, ao ler o paper do VICTOR, um dos problemas já citados é que esse dataset é um **conjunto de dados desbalanceado**. Por isso, também vou realizar o **estudo de técnicas** para lidar com esse tipo de dataset e técnicas para criação de dados sintéticos caso seja necessário, como a técnica SMOTE (Synthetic Minority Over-sampling Technique), por exemplo. Além disso, irei **definir o pipeline** a ser seguido na implementação prática e os **frameworks** a serem utilizados (utilizando de referência o levantamento realizado essa semana).

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

[Documento citado no Termo de Aceite de Entrega de 16 de novembro]

Estudo de frameworks

Esse documento contém um levantamento dos principais frameworks para a tarefa de classificação de textos. Foram consideradas as etapas listadas abaixo de desenvolvimento de um modelo de classificação de textos. Dependendo da implementação pode ser que algumas etapas não sejam adotadas ou que outras não listadas sejam, porém de forma geral as etapas a serem consideradas são as descritas abaixo:

- Importação dos dados
- Pré-processamento / feature engineering
- Feature extraction
- Treino do modelo
- Obtenção de métricas
- Otimização de hiperparâmetros

Vale lembrar que todos os frameworks comentados abaixo são da linguagem Python.

Pandas: Oferece estruturas de dados poderosas, como DataFrames, que simplificam a manipulação e transformação de conjuntos de dados. Especificamente, no contexto de processamento de linguagem natural (NLP), o Pandas é frequentemente utilizado para a importação, pré-processamento e manipulação de dados textuais. Suas funcionalidades incluem operações eficientes para lidar com texto, como remoção de duplicatas, seleção de features e engenharia de características (feature engineering). Apesar de ser uma ferramenta de alto desempenho em tarefas de manipulação de dados, é importante notar que para volumes muito grandes de dados, o Pandas pode exigir uma quantidade considerável de memória. https://pandas.pydata.org/

PySpark: Desenvolvido como uma interface Python para o Apache Spark, o PySpark é uma biblioteca que oferece poderosas capacidades de processamento distribuído. Projetado

para lidar com grandes volumes de dados, é especialmente valioso em ambientes de big data, onde o Spark é frequentemente utilizado. No contexto do processamento de linguagem natural (NLP), o PySpark é aplicado para tarefas de importação de dados, pré-processamento e análise em larga escala. Sua arquitetura distribuída permite a execução eficiente de operações em clusters, tornando-o adequado para lidar com conjuntos de dados extensos e complexos em projetos de NLP em grande escala. Possui funcionalidade similares ao Pandas, porém ao contrário dele, que é mais eficiente para operações em conjuntos de dados que cabem na memória, o PySpark é projetado para processamento distribuído, tornando-o uma escolha poderosa em ambientes onde o dimensionamento é essencial. https://spark.apache.org/docs/latest/api/python/index.html

NLTK (Natural Language Toolkit): Capaz de auxiliar em diversas tarefas de pré-processamento textual, oferece uma ampla variedade de módulos para tokenização, lematização, análise sintática, entre outras operações. Sua versatilidade é evidente na capacidade de lidar com uma variedade de tarefas, desde simples manipulações de texto até análises linguísticas mais complexas. Além disso, o NLTK inclui recursos como corpora e modelos pré-treinados, facilitando a implementação de soluções em projetos de NLP. https://www.nltk.org/

VADER: Valence Aware Dictionary and sEntiment Reasoner (VADER) é uma biblioteca especificamente projetada para análise de sentimento em mídias sociais e inclui uma abordagem baseada em léxico (lexicon based) que é ajustada para a linguagem das mídias sociais. Inclui um léxico de sentimento pré-construído com medidas de intensidade para sentimento positivo e negativo, e incorpora regras para lidar com intensificadores de sentimento, emojis e outros recursos específicos que, via de regra, são utilizados em mídias sociais. https://github.com/cjhutto/vaderSentiment. Possui uma adaptação para português: https://github.com/rafjaa/LeIA

SpaCy: Oferece uma ampla gama de funcionalidades, desde tokenização e lematização até análise sintática e reconhecimento de entidades nomeadas. No âmbito do processamento de linguagem natural, o SpaCy é frequentemente utilizado para realizar diversas etapas do pré-processamento textual. Ele se destaca por sua capacidade de processar grandes volumes de texto de forma rápida e eficiente. Além disso, o SpaCy possui modelos pré-treinados para várias línguas, o que facilita a incorporação de recursos linguísticos em projetos de NLP. Uma característica notável do SpaCy é a sua extensibilidade, permitindo que os usuários adicionem componentes personalizados para atender a requisitos específicos. Com documentação abrangente e uma comunidade ativa, o SpaCy é uma escolha popular em projetos que envolvem análise de texto e processamento de linguagem natural. https://spacy.io/

TextBlob: É uma biblioteca em Python que oferece uma gama de ferramentas para tarefas de pré-processamento e análise de texto. Embora possua menos recursos em comparação com outros frameworks, é mais recomendado para projetos que buscam soluções mais simples e eficazes. Facilita tarefas comuns de NLP, como tokenização, lematização e análise de sentimentos, possuindo uma interface mais simples que permite a extração rápida de informações relevantes de texto, enquanto a funcionalidade de análise de sentimentos incorporada oferece uma solução direta para avaliar a polaridade e subjetividade de expressões. Embora não seja tão extensivo quanto alguns frameworks especializados em NLP, o TextBlob é uma escolha prática para projetos nos quais a facilidade de implementação é valorizada. https://textblob.readthedocs.io/en/dev/

Scikit-learn: É uma ferramenta essencial para tarefas de aprendizado de máquina em Python, destacando-se por sua robustez e variedade de algoritmos para classificação, regressão, clustering e, especialmente, e pelo suporte sólido às tarefas de Processamento de Linguagem Natural (NLP). Ao integrar funcionalidades avançadas, o Scikit-learn oferece não apenas algoritmos de aprendizado de máquina, mas também ferramentas essenciais para

pré-processamento de dados textuais. O Scikit-learn se destaca por sua capacidade de realizar extração de características, incluindo técnicas como Bag-of-Words (BoW) e Term Frequency-Inverse Document Frequency (TF-IDF). Além disso, o framework facilita a implementação de pipelines completos para tarefas complexas de NLP, desde a tokenização até a criação de modelos preditivos, e possui um design modular e documentação abrangente. Também possui funcionalidades para avaliar o desempenho de modelos, como F1 e acurácia. https://scikit-learn.org/stable/

FastText: É uma biblioteca que se destaca pela capacidade de treinar modelos de embeddings de palavras e realizar classificação de texto de forma rápida e eficaz. Desenvolvido pelo Facebook, este framework aprende representações vetoriais de palavras e também leva em consideração a estrutura de subpalavra (subword) das palavras. É projetado para lidar com grandes conjuntos de dados textuais e oferece uma interface fácil de usar para tarefas comuns de NLP, como classificação de texto e análise de sentimentos. Além disso, sua funcionalidade de embeddings de palavras pré-treinados permite incorporar facilmente conhecimento linguístico em seus modelos. Dessa forma, o FastText se destaca como uma escolha sólida para projetos de NLP que demandam velocidade e desempenho. https://fasttext.cc/

Gensim: É especialmente destacado por sua eficiência na modelagem de tópicos e na criação de representações semânticas de documentos. Projetado para lidar com grandes conjuntos de dados textuais, o Gensim oferece uma implementação eficaz de algoritmos de modelagem de tópicos, como o Latent Dirichlet Allocation (LDA), permitindo a descoberta de padrões latentes em grandes coleções de documentos. Uma característica distintiva do Gensim é sua capacidade de treinar modelos de embeddings de palavras, como o Word2Vec e Glove. Além disso, o Gensim oferece funcionalidades para similaridade de documentos, indexação eficiente e manipulação de grandes quantidades de texto de maneira escalável.Ou

seja, é um framework recomendado para fazer o uso de embeddings estáticos, como Word2Vec. https://radimrehurek.com/gensim/

Hugging Face: Reconhecido como um hub central para inovações em processamento de linguagem natural (NLP),também é uma plataforma que oferece uma ampla gama de modelos pré-treinados, datasets e ferramentas para tarefas diversas em NLP. Destaca-se por sua comunidade ativa e pela capacidade de compartilhar, descobrir e implementar modelos de última geração. Uma de suas características marcantes é a biblioteca Transformers, que facilita o acesso e o uso de uma variedade de modelos pré-treinados de última geração, como BERT, GPT-3 e muitos outros. Essa abordagem simplifica significativamente a implementação de modelos de NLP avançados, estimulando a pesquisa e o desenvolvimento em larga escala. Assim, é a principal ferramenta para fazer uso de modelos baseados em Transformers, além de possuir funções para treino, fine-tunning e avaliação dos modelos. https://huggingface.co/

TensorFlow: Como uma das principais bibliotecas para aprendizado de máquina e processamento de linguagem natural (NLP), o TensorFlow se destaca por sua versatilidade e eficiência computacional. Desenvolvido pela Google, o TensorFlow oferece uma infraestrutura robusta para a criação e treinamento de modelos de aprendizado profundo, sendo amplamente adotado em projetos de NLP. Para tarefas específicas de NLP, o TensorFlow disponibiliza módulos e ferramentas, incluindo o TensorFlow Text, que oferece funcionalidades avançadas para processamento de texto, como tokenização, embeddings de palavras e camadas especializadas para tarefas como classificação de texto e tradução. A integração natural do TensorFlow com unidades de processamento gráfico (GPU) e suas capacidades de computação distribuída fazem dele uma escolha poderosa para treinamento de modelos em larga escala. Além disso, a comunidade ativa e a extensa documentação do TensorFlow contribuem para sua popularidade entre pesquisadores e desenvolvedores de NLP. https://www.tensorflow.org/?hl=pt-br

PyTorch: Similar ao TensorFlow, destaca-se como um dos principais frameworks para aprendizado de máquina e, em particular, para processamento de linguagem natural (NLP), o PyTorch é reconhecido por sua flexibilidade e interface intuitiva. Desenvolvido pelo Facebook, o PyTorch oferece uma abordagem dinâmica e orientada para o usuário, o que facilita a construção e experimentação rápida com modelos complexos. Para tarefas específicas de NLP, o PyTorch conta com o pacote torchtext, que simplifica o pré-processamento de dados textuais e fornece utilitários para carregamento eficiente de conjuntos de dados. Além disso, o PyTorch é a escolha preferida para muitas pesquisas e implementações de modelos de linguagem, especialmente em contextos nos quais a exploração de arquiteturas personalizadas é crucial. Sua integração natural com a computação em GPU e o foco na experiência do usuário fazem dele uma ferramenta valiosa para projetos que demandam flexibilidade e eficácia no desenvolvimento de modelos de linguagem. https://pytorch.org/

MXNet: Destacando-se como uma biblioteca de aprendizado profundo escalável e eficiente, é especialmente reconhecido por sua arquitetura flexível e suporte eficaz para treinamento distribuído. Desenvolvido pela Apache Software Foundation, oferece uma plataforma robusta para projetos de processamento de linguagem natural (NLP) e outras tarefas de aprendizado de máquina. Com interfaces para Python e outras linguagens, o MXNet é acessível e versátil, sendo possível ao usuário montar suas próprias redes neurais. Além disso, é conhecido por sua eficiência em treinamento de modelos em ambientes distribuídos, tornando-o adequado para lidar com grandes volumes de dados. Seja para experimentação em pequena escala ou implementações em larga escala, o MXNet oferece uma variedade de recursos para modelagem e treinamento de redes neurais em projetos que envolvem NLP e outras tarefas complexas de aprendizado de máquina. https://mxnet.apache.org/

FastAI: Reconhecido por tornar o aprendizado profundo mais acessível, o FastAI é uma biblioteca construída sobre o PyTorch, que fornece uma interface de alto nível que simplifica o processo de criação e treinamento de modelos complexos. O FastAI oferece abstrações poderosas para tarefas comuns em aprendizado profundo, incluindo processamento de linguagem natural (NLP). Para tarefas específicas de NLP, ele conta com o módulo fastai.text, que simplifica o pré-processamento de texto, o carregamento de dados e a criação de modelos de linguagem. Além disso, a biblioteca inclui funcionalidades avançadas, como transferência de aprendizado e métodos de treinamento eficazes, que são especialmente úteis para experimentação rápida e desenvolvimento de modelos de NLP com desempenho superior. https://github.com/fastai/fastai

Ray Tune: Especializado em otimização de hiperparâmetros, é uma biblioteca eficiente para busca e ajuste sistemático de configurações ideais de modelos. Desenvolvido sobre o ecossistema Ray, oferece uma interface intuitiva para experimentação escalável e distribuída. Ideal para projetos de processamento de linguagem natural (NLP) e aprendizado de máquina. Além disso, simplifica a seleção de hiperparâmetros, acelerando a descoberta de configurações otimizadas e melhorando o desempenho dos modelos. https://docs.ray.io/en/latest/tune/

Optuna: Similar ao Ray Tune, é focado em otimização de hiperparâmetros eficiente e automática, e oferece uma abordagem flexível para encontrar as melhores configurações de modelos. Faz uso de algoritmos de busca eficazes para explorar o espaço de hiperparâmetros, acelerando a descoberta de configurações otimizadas, além de possuir suporte para integração em diversas bibliotecas de aprendizado de máquina, como TensorFlow, PyTorch e Scikit-learn. https://optuna.readthedocs.io/en/stable/

WandB (Weights & Biases): Reconhecido como uma plataforma abrangente para rastreamento, visualização e colaboração em projetos de aprendizado de máquina, oferece

uma solução completa para experimentação e monitoramento de modelos. Com suporte para várias bibliotecas, como TensorFlow e PyTorch, ele simplifica a análise e compartilhamento de resultados. Para projetos de processamento de linguagem natural (NLP) e outras tarefas, o WandB permite o registro de métricas, gráficos interativos e visualização de embeddings, fornecendo uma compreensão aprofundada do desempenho do modelo. Além disso, sua integração com fluxos de trabalho de aprendizado profundo facilita a compreensão e otimização contínua dos modelos. Ou seja, permite uma análise detalhada e compartilhamento transparente de experimentos. https://wandb.ai/

APÊNDICE 3

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 23 de nov. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Héber Júnior

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Conforme comentado na última semana, para a entrega atual realizei uma análise exploratória do dataset VICTOR (dataset usado para classificação de textos jurídicos). A análise pode ser encontrada nesse notebook: Análise explorátoria dataset VICTOR.ipynb

Essa análise exploratória permitiu observar alguns pontos interessantes. O primeiro ponto é que podemos perceber o grande desbalanceamento do dataset, podemos também observar que a maioria das sentenças são bem extensas, mas ao mesmo tempo respeitam o limite de tokens dos modelos baseados na arquitetura transformers, que é de 512 tokens. Por serem textos longos, podemos imaginar que os modelos baseados em contextos irão obter resultados mais interessantes nesse dataset, o que será testado nas próximas semanas.

Também foi feita uma análise visando encontrar as palavras mais importantes para cada uma das classes. Além disso, foi constatado uma certa semelhança entre as palavras mais comuns em cada classe, o que pode ter implicações interessantes para a escolha de modelos e representações.

Além dessa análise exploratória, também foi feito o estudo de técnicas para criar dados sintéticos e técnicas para trabalhar com conjuntos de dados desbalanceados. O estudo pode ser encontrado nesse link: Gate 23-11

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Para a próxima semana, irei começar com os experimentos práticos da minha área de aplicação. A ideia é seguir o pipeline de uma tarefa de classificação de textos trabalhando inicialmente com representações e classificadores clássicos, como o TF-IDF para representar

as palavras e algoritmos como o SVM e regressão linear para a parte de classificação.

O pipeline a ser seguido será o seguinte:

- Importação dos dados
- Realizar pré-processamento mais avançado nos textos, como tokenização, remoção de stopwords.
- Extração de características / representação dos textos com métodos clássicos (TF-IDF)
- Uso de algoritmos classificadores clássicos (regressão logística)
- Avaliação dos resultados (uso de métricas adequadas para o problema)

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

[Documento citado no Termo de Aceite de Entrega de 23 de novembro]

Dados sintéticos e dataset desbalanceado

Esse documento contém o resumo do estudo sobre as técnicas usadas para criar dados sintéticos e técnicas para lidar com conjuntos desbalanceados, o que é muito comum em textos da área jurídica.

Artigo 1: Data Augmentation: Can BERT Do The Work For You?

https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1214/reports/final_reports/report254.pdf

O artigo aborda a eficácia do data augmentation automático por meio de modelos de linguagem, como o BERT, para melhorar a robustez de modelos neurais. Ele compara dois métodos de augmentation: o "bert context rewriter" que utiliza o modelo de linguagem mascarado DistilBERT, e o "token reorderer" que usa o conhecimento semântico do Universal Sentence Encoder.

Token Reorderer: Reordena tokens em perguntas com base na importância determinada pela similaridade entre as sentenças original e modificada usando o Universal Sentence Encoder.

Bert Context Rewriter: Utiliza o modelo de linguagem mascarado DistilBERT para reescrever contextos em tarefas de question answering, substituindo tokens mascarados pelos resultados mais confiantes do modelo.

Artigo 2: Data Augmentation Using Pre-trained Transformer Models

https://aclanthology.org/2020.lifelongnlp-1.3.pdf

Este artigo explora o uso de modelos de linguagem pré-treinados baseados em transformers, como BERT e GPT-2, para realizar aumento de dados em tarefas de processamento de linguagem natural (PLN). O objetivo é abordar desafios associados a

métodos tradicionais de substituição de palavras, que muitas vezes não preservam efetivamente rótulos de classes. Os autores propõem uma abordagem unificada, demonstrando que condicionar esses modelos à inclusão de rótulos de classes nas sequências de texto é uma maneira eficaz de realizar aumento de dados.

Além dessas técnicas baseadas em arquiteturas transformers para criar dados sintéticos, temos técnicas mais simples que dependendo do problema podem ser igualmente úteis, são elas:

Retrotradução: primeiro convertemos a frase para um idioma diferente usando um modelo e depois convertemos novamente para o idioma alvo. Como usamos um modelo de ML para isso, resulta em uma frase equivalente à frase original, mas com palavras diferentes. Existem vários modelos pré-treinados disponíveis como Google T5 , Facebook NMT (Neural Machine Translation), etc.

Inserção Aleatória: nessa técnica, inserimos aleatoriamente uma palavra em uma determinada frase. Uma maneira é inserir qualquer palavra aleatoriamente, mas também podemos usar modelos pré-treinados como BERT para inserir uma palavra com base no contexto.

Substituição Aleatória: nessa técnica substituímos uma palavra aleatória por uma nova palavra, podemos usar dicionários pré-construídos para substituir por sinônimos ou podemos usar modelos pré-treinados como BERT .

Geração de Texto: Nesta técnica utilizamos modelos generativos como GPT3, distilgpt2. Alimentamos o texto original como início e então o modelo gera palavras adicionais com base no texto de entrada, desta forma adicionamos ruído aleatório à nossa frase.

Links de referência:

[1] NLP Data Augmentation using Transformers, https://towardsdatascience.com/nlp-data-augmentation-using-transformers-89a44a993bab

[2] 5 Data Augmentation Techniques for Text Classification, https://saurabhk30.medium.com/5-data-augmentation-techniques-for-text-classification-d14f6d8bd6aa

Outro ponto interessante também é saber como lidar com o conjunto de dados desbalanceado sem a necessidade de criar dados sintéticos. Abaixo listo algumas técnicas que possibilitam isso:

Subamostragem (Undersampling): Remover aleatoriamente exemplos da classe majoritária para equilibrar o conjunto de dados. Isso pode ser útil quando temos uma classe majoritária com uma quantidade de exemplos muito grande e reduzir os exemplos dessa classe não será prejudicial para o desempenho do modelo.

Peso de Classe: Atribuir pesos diferentes às classes durante o treinamento para dar mais importância à classe minoritária. Isso pode ser feito em muitos algoritmos de aprendizado de máquina, incluindo SVM, Random Forests e redes neurais.

Métricas de Avaliação Adequadas: podemos usar métricas de avaliação que sejam sensíveis ao desbalanceamento, como precisão, recall, F1-score. Em classificação de textos usamos bastante a métrica F1-score pois ela consegue levar em consideração o desbalanceamento do dataset.

APÊNDICE 4

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: **30 de nov. de 2023**

Participantes da Entrega **[matriculados em Residência em IA]:**

Héber Júnior

Entrega: **[descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]**

Conforme mencionado no planejamento da última semana, nessa entrega comecei a realização dos experimentos da tarefa de classificação de textos jurídicos. Por isso, segui todo o pipeline necessário:

- Importação dos dados
- Realização do pré-processamento nos textos, remoção de stopwords, lowercase, remoção de caracteres especiais, etc.
- Extração de características / representação dos textos com métodos clássicos (TF-IDF)
- Uso de algoritmos classificadores clássicos
- Avaliação dos resultados (uso de métricas adequadas para o problema, como F1-score)

Nos experimentos foram avaliados os desempenhos COM e SEM o uso de técnicas de pré-processamento nos textos, onde os resultados com pré-processamento mostraram um melhor desempenho. Além disso, também foi avaliado o desempenho da representação TF-IDF nesse caso de textos jurídicos que são bastante extensos. Apesar de resultados interessantes para um baseline, acredito que as representações baseadas em contexto poderão apresentar um melhor resultado devido a característica dos textos jurídicos e isso será validado nos próximos experimentos.

A descrição de tudo que foi feito e os códigos podem ser encontrados nesse link:Gate 30/11/23

Planejamento: **[descrever o que pretende fazer para realizar a próxima ENTREGA]**

Para a próxima semana, irei realizar mais experimentos práticos no dataset VICTOR. A ideia é trabalhar com representações mais robustas, como as representações baseadas na arquitetura transformers que conseguem lidar melhor com o contexto.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: **Em análise!**

[Documento citado no Termo de Aceite de Entrega de 30 de novembro]

Experimentos com uso de técnicas clássicas

Estou trabalhando com Processamento de Linguagem Natural (NLP), mais especificamente com processamento de textos jurídicos. Para a parte prática da residência, estou focado na parte de classificação de textos.

Na última semana, foi feita uma análise exploratória do dataset VICTOR (dataset para classificação de textos jurídicos do STF). Nessa semana, segui trabalhando com esse dataset, mas agora realizando todo o pipeline de uma tarefa de classificação de textos:

- Importação dos dados
- Realização do pré-processamento nos textos, remoção de stopwords, lowercase, remoção de caracteres especiais, etc.
- Extração de características / representação dos textos com métodos clássicos (TF-IDF)
- Uso de algoritmos classificadores clássicos
- Avaliação dos resultados (uso de métricas adequadas para o problema, como F1-score)

Nesse experimento foi utilizada a representação de palavras conhecida como TF-IDF (Term Frequency-Inverse Document Frequency), essa abordagem ajuda a capturar a importância relativa das palavras em um conjunto de documentos.

Primeiro foi feita a importação do dataset e a filtragem das colunas que serão utilizadas, que no caso são as colunas “body” e a coluna “document_type”. A coluna “body” contém o texto e a coluna “document_type” diz qual é a classe que representa cada texto, o que caracteriza uma abordagem supervisionada.

Após isso, foi feita verificação se existiam valores vazios no dataset e também foram excluídos os dados duplicados que poderiam causar prejuízo durante o treinamento dos modelos.

Verificando a distribuição das classes, percebi um grande desbalanceamento entre elas. Como a classe majoritária (‘outros’) possui muito mais amostras que a soma de todas as outras classes, a estratégia abordada foi a técnica chamada Undersampling. Nesse caso foram removidos alguns exemplos da classe majoritária sem prejudicar o desempenho do modelo.

Nos trabalhos que lidam com textos jurídicos é comum realizarmos várias etapas de pré-processamento nos textos para obtermos melhores resultados. Algumas etapas seguidas são destacadas no artigo “Preprocessing Applied to Legal Text Mining: analysis andevaluation of the main techniques used” e são elas:

- Remoção de stopwords
- Remoção de acentos das palavras
- Colocar todas as palavras em lowercase (em letra minúscula)
- Remoção de caracteres especiais (@, &, $, por exemplo).

Link do artigo citado acima:
(https://sol.sbc.org.br/index.php/eniac/article/view/25760/25576)

Foram realizados experimentos tanto utilizando pré-processamento nos textos, como também sem a utilização do pré-processamento. Foi possível notar a diferença nos resultados dessas duas abordagens. O uso do pré-processamento traz sim ganhos para a classificação de textos jurídicos. Algumas classes tiveram uma leve queda de desempenho, mas nada significativo. Já outras classes tiveram uma melhora bastante significativa e de forma geral, o modelo consegue apresentar um equilíbrio melhor no resultado quando usamos os textos com os pré-processamentos citados anteriormente.

Os experimentos com e sem pré-processamento foram todos realizados sob mesma condição: O conjunto de dados foi dividido em treino, validação e teste. A forma de representação de palavras utilizada foi o TF-IDF e o algoritmo classificador foi o de árvore de decisão.

Desempenho do modelo sem pré-processamento:

Desempenho no conjunto de teste:

	precision	recall	f1-score	support
outros	0.87	0.88	0.87	4241
peticao_do_RE	0.61	0.58	0.59	949
agravo_em_recurso_extraordinario	0.59	0.61	0.60	395
sentenca	0.59	0.60	0.60	278
acordao_de_2_instancia	0.47	0.44	0.45	66
despacho_de_admissibilidade	0.27	0.33	0.30	36
accuracy			0.79	5965
macro avg	0.57	0.57	0.57	5965
weighted avg	0.79	0.79	0.79	5965

Desempenho do modelo com pré-processamento:

Desempenho no conjunto de teste:

	precision	recall	f1-score	support
outros	0.87	0.87	0.87	4241
peticao_do_RE	0.59	0.57	0.58	949
agravo_em_recurso_extraordinario	0.53	0.56	0.55	395
sentenca	0.52	0.50	0.51	278
acordao_de_2_instancia	0.45	0.38	0.41	66
despacho_de_admissibilidade	0.36	0.44	0.40	36
accuracy			0.78	5965
macro avg	0.55	0.55	0.55	5965
weighted avg	0.78	0.78	0.78	5965

De forma geral, a etapa de pré-processamento deu uma certa equilibrada entre as classes. Além disso, por se tratar de textos jurídicos (que são longos) o contexto é algo muito importante a se considerar. A representação de TF-IDF como um baseline foi interessante, mas como já foi citado anteriormente, esse dataset não tem tantas palavras específicas que podem diferenciar uma classe das demais, o que poderia favorecer representações com a TF-IDF. Por isso, é esperado que as representações baseadas na arquitetura transformers possam trazer melhores resultados por lidarem bem com contexto e isso será validado nas próximas semanas.

Link do colab com os experimentos que foram feitos: Gate6-30/11.ipynb

APÊNDICE 5

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 7 de dez. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Héber Júnior

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Conforme planejado na última semana, para essa entrega segui o pipeline de classificação de textos usando representações baseadas na arquitetura Transformers. Para isso, segui o pipeline:

- Importação dos dados
- Realização do pré-processamento nos textos, remoção de stopwords, lowercase, remoção de caracteres especiais, etc.
- Extração de características / representação dos textos com métodos baseados na arquitetura transformers (RoBERTa)
- Avaliação dos resultados (uso de métricas adequadas para o problema, como F1-score)

Nos experimentos foi possível validar que as representações baseadas na arquitetura transformers conseguem trazer um resultado melhor se comparado com as abordagens clássicas, como TF-IDF (quando utilizamos o dataset VICTOR). Isso se dá por alguns fatores, entre eles, porque essas representações conseguem lidar melhor com textos longos e com o contexto, algo que se faz muito necessário quando falamos de textos jurídicos.

A descrição dos experimentos feitos e os códigos podem ser encontrados nesse link: Gate 07/12/23

Planejamento: **[descrever o que pretende fazer para realizar a próxima ENTREGA]**

Para a próxima semana, irei trabalhar com a sumarização dos textos jurídicos. Considerando que são textos longos e utilizar modelos com representações Transformers exige um custo computacional maior, sumarizar os textos pode reduzir o tempo de treinamento, otimizar o treino dos modelos.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: **Em análise!**

[Documento citado no Termo de Aceite de Entrega de 07 de dezembro]

Experimentos com uso de modelos baseados na arquitetura Transformers

Durante a residência estou trabalhando com Processamento de Linguagem Natural (NLP), mais especificamente com processamento de textos jurídicos. Atualmente estou trabalhando na tarefa de classificação de textos.

Sigo trabalhando com o dataset VICTOR (dataset para classificação de textos jurídicos). Na entrega anterior, realizei experimentos práticos utilizando abordagens clássicas como TF-IDF (Term Frequency-Inverse Document Frequency) para representação de palavras.

Para essa entrega, estou trabalhando com as representações baseadas na arquitetura transformers, são representações mais ricas e que conseguem lidar bem com contexto e textos longos (que é o caso dos textos jurídicos). Como a ideia é comparar as diferentes representações, o pipeline seguido será parecido com o utilizado na representação TF-IDF:

- Importação dos dados
- Realização do pré-processamento nos textos, remoção de stopwords, lowercase, remoção de caracteres especiais, etc.
- Extração de características / representação dos textos com métodos baseados na arquitetura transformers (RoBERTa)
- Avaliação dos resultados (uso de métricas adequadas para o problema, como F1-score)

Nesses experimentos, foram feitas a verificação de valores ausentes e a exclusão de dados duplicados do dataset. Por conta do desbalanceamento entre as classes, a técnica usada para contornar esse problema foi o uso de undersampling (foram removidos alguns exemplos da classe majoritária sem prejudicar o desempenho do modelo).

Além disso, os pré-processamentos realizados nos textos foram:

- Remoção de stopwords
- Remoção de acentos das palavras
- Colocar todas as palavras em lowercase (em letra minúscula)
- Remoção de caracteres especiais (@, &, $, por exemplo).

Artigo de referência para o pré-processamento:
https://sol.sbc.org.br/index.php/eniac/article/view/25760/25576

Também foram utilizadas as mesmas divisões para treino, validação e teste para que a comparação com as representações TF-IDF fosse justa.

De forma geral, pode-se perceber uma melhora significativa ao utilizar representações baseadas na arquitetura transformer, como a RoBERTa. Alguns motivos que podem ajudar a arquitetura transformers entregar um resultado superior:

Capacidade de lidar com sequências longas: Transformers são projetados para lidar eficientemente com sequências longas, capturando dependências de longo alcance. Isso é importante em textos jurídicos, que frequentemente envolvem documentos extensos com informações relevantes distribuídas ao longo do texto.

Transferência de Aprendizado: Modelos Transformers geralmente são pré-treinados em grandes corpos de texto, ganhando um conhecimento prévio amplo sobre a linguagem. Esse conhecimento prévio é transferido para tarefas específicas durante o ajuste fino (fine-tuning), proporcionando uma vantagem significativa, especialmente quando os dados de treinamento específicos são limitados.

Contexto global e representações contextualizadas: Transformers, como RoBERTa, capturam relações semânticas complexas e contextualizadas em todo o texto. Eles consideram o contexto global das palavras em uma sentença, permitindo a captura de dependências contextuais que métodos baseados em Bag of Words (como TF-IDF) podem perder.

Além disso, outros fatores devem ser considerados quando falamos do uso de arquiteturas transformers:

São representações mais robustas que as representações TF-IDF que conseguem lidar melhor com contexto e conseguem representar os textos de uma maneira mais rica, mas tudo isso tem um custo. O tempo de treinamento é maior e necessita de mais recursos computacionais.

Para lidar com esses problemas, nas próximas semanas irei utilizar algumas estratégias. A primeira delas será trabalhar com a sumarização dos textos. A ideia é adicionar a sumarização na etapa de pré-processamento, assim terei textos menores, o que pode reduzir o tempo de treinamento. Além disso, nas próximas semanas também irei trabalhar com outras formas de otimização para a arquitetura transformers.

Como já citado anteriormente, os resultados conseguem ser superiores ao utilizar modelos dessa natureza, mas eles também demandam mais recursos computacionais e nas próximas semanas irei trabalhar com formas de otimizar esses modelos, tanto a nível de treinamento quanto a nível de inferência.

A imagem abaixo é o resultado do experimento feito utilizando a representação RoBERTa, o modelo foi treinado por 2 épocas.

```
Epoch 1
Training loss: 0.6214707145952199
Validation loss: 0.4251231153635946
Validation F1 Score (Weighted): 0.8438411912702292
Test loss: 0.41610771167564065
Test F1 Score (Weighted): 0.8466556312342731

Epoch 2
Training loss: 0.3428285017150314
Validation loss: 0.35589679560738885
Validation F1 Score (Weighted): 0.87326971984343
Test loss: 0.33892829446504785
Test F1 Score (Weighted): 0.8781142257558214
```

No notebook (colab) pode ser verificado todos os passos seguidos durante a realização desse experimento: Héber - Gate7-07/12.ipynb

APÊNDICE 6

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: **14 de dez. de 2023**

Participantes da Entrega **[matriculados em Residência em IA]**:

Héber Júnior

Entrega: **[descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]**

Conforme planejado na última semana, nessa entrega trabalhei com a sumarização dos textos jurídicos. A sumarização por si só é uma tarefa muito importante quando falamos de textos longos e já resolve parte dos problemas no mundo do direito. Além disso, a sumarização também pode ser usada como uma etapa no pipeline de classificação de textos (pode ser usada na parte de pré-processamento dos textos visando ter textos mais enxutos).

Por isso, realizei experimentos com o uso da sumarização de duas formas:

- Sumarizando parte do dataset (em casos onde os textos passavam do limite de 512 tokens
- Sumarizando o dataset inteiro e comparando com experimentos sem realizar a sumarização

A descrição dos experimentos que foram feitos pode ser encontrada no seguinte link:Gate 14/12/2023

Planejamento: **[descrever o que pretende fazer para realizar a próxima ENTREGA]**

Para a próxima entrega, o objetivo é trabalhar com formas de otimizar os modelos

classificadores que usam a arquitetura transformers. Esses são modelos pesados e podem ser otimizados usando técnicas como quantização e otimização em grafo, visando poder realizar a inferência/colocar os modelos em produção usando CPUs e não sendo necessário o uso de GPUs. Isso reduz os custos e também o tempo de processamento.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

[Documento citado no Termo de Aceite de Entrega de 14 de dezembro]

Sumarização dos textos

Estou trabalhando com Processamento de Linguagem Natural (NLP) focado no processamento de textos jurídicos. Nas duas últimas semanas trabalhei na tarefa de classificação de textos usando o dataset VICTOR, primeiro usando abordagens clássicas como TF-IDF e depois trabalhei com abordagens que usam arquitetura transformers, como RoBERTa.

Pensando em ser um especialista no processamento de textos do domínio jurídico, é importante conhecer outras tarefas que podem ser feitas nessa área. Nesta semana trabalhei com a tarefa de sumarização de textos. Os textos jurídicos têm características próprias e uma delas é de serem textos bem extensos. Por isso, a tarefa de sumarizar os textos se faz muito necessária.

Sumarizar por si só já é uma tarefa que pode resolver boa parte dos problemas no ambiente jurídico e além disso, essa tarefa também pode ser aplicada no pipeline de classificação de textos (podemos incluir a sumarização no pré-processamento de textos para depois treinarmos os classificadores). De uma forma simples, ao sumarizar queremos ter um resumo de um texto que era longo, sem perder as principais informações.

O modelo usado para sumarização é baseado no T5 do Google (com fine-tuned em textos jurídicos) e pode ser encontrado em:

stjiris/t5-portuguese-legal-summarization · Hugging Face

Um artigo interessante que embasa o uso do modelo T5 é o “Abstractive Text Summarization using Transfer Learning” que pode ser encontrado em:

Abstractive Text Summarization using Transfer Learning

Este artigo explora o problema da sumarização de texto abstrativo. O estudo mostra que o modelo baseado em Transfer Learning alcançou melhorias significativas na

sumarização de texto abstrativo, superando abordagens tradicionais. Além disso, destaca o desempenho positivo do modelo T5 na sumarização.

Os experimentos foram feitos usando duas abordagens:

Na primeira abordagem, foi utilizado o dataset completo (o mesmo utilizado nos experimentos do gate anterior). Nesse caso, foram sumarizados somente os textos que passavam de 512 tokens. Vale lembrar que modelos como o BERT tem um limite de 512 tokens, então os textos maiores que isso seriam truncados.

Ao usar esse limiar de sumarizar textos maiores que 512 tokens, cerca de 600 amostras foram sumarizadas e o mais interessante é que os resultados (a métrica utilizada é a F1-score na classificação de textos) não foram prejudicados e podem ser visualizados abaixo:

Figura 1: experimento sem realizar nenhum tipo de sumarização, usando o dataset original:

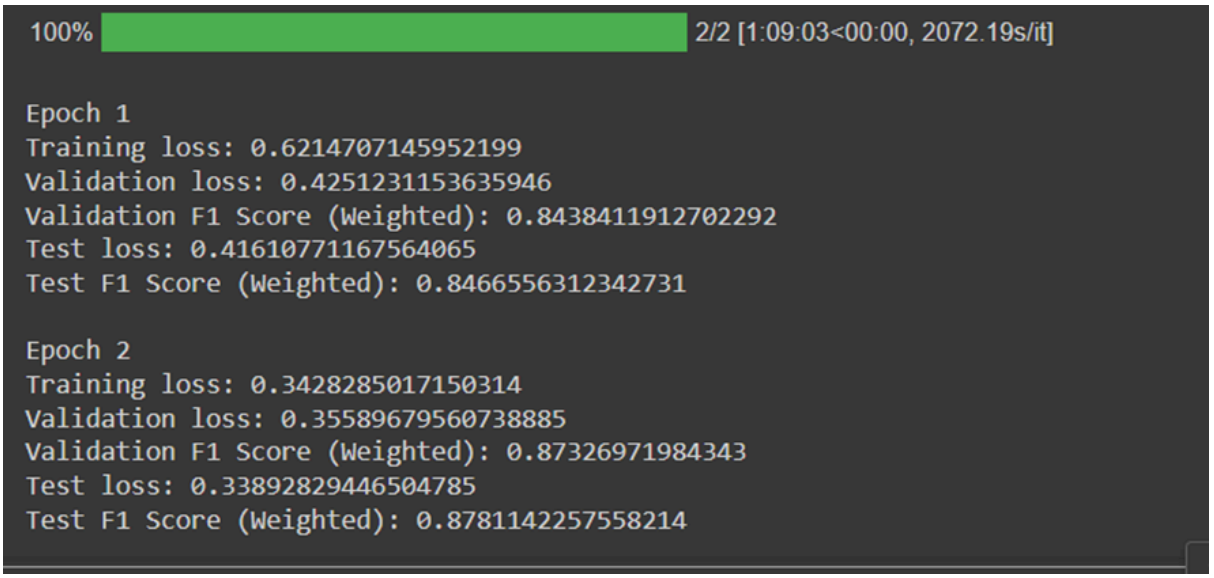


Figura 2: experimento usando o mesmo dataset da figura 1, porém sumarizando os textos maiores que 512 tokens

```
100% ████████████████████████████████ 2/2 [1:07:07<00:00, 2013.99s/it]

Epoch 1
Training loss: 0.6369063967507179
Validation loss: 0.4577905219915795
Validation F1 Score (Weighted): 0.8372618094357144
Test loss: 0.42739146143606266
Test F1 Score (Weighted): 0.8433679430338357

Epoch 2
Training loss: 0.3479847161141841
Validation loss: 0.3661748809204118
Validation F1 Score (Weighted): 0.8748333132083256
Test loss: 0.35027482146269656
Test F1 Score (Weighted): 0.880655277384151
```

Vale ressaltar que nesse primeiro caso os textos foram sumarizados para respeitar o limite de 512 tokens.

Na segunda abordagem, foi escolhida uma amostra do dataset com cerca de mil exemplos. O objetivo deste experimento é comparar o desempenho sem sumarização e com sumarização de 100% dos dados.

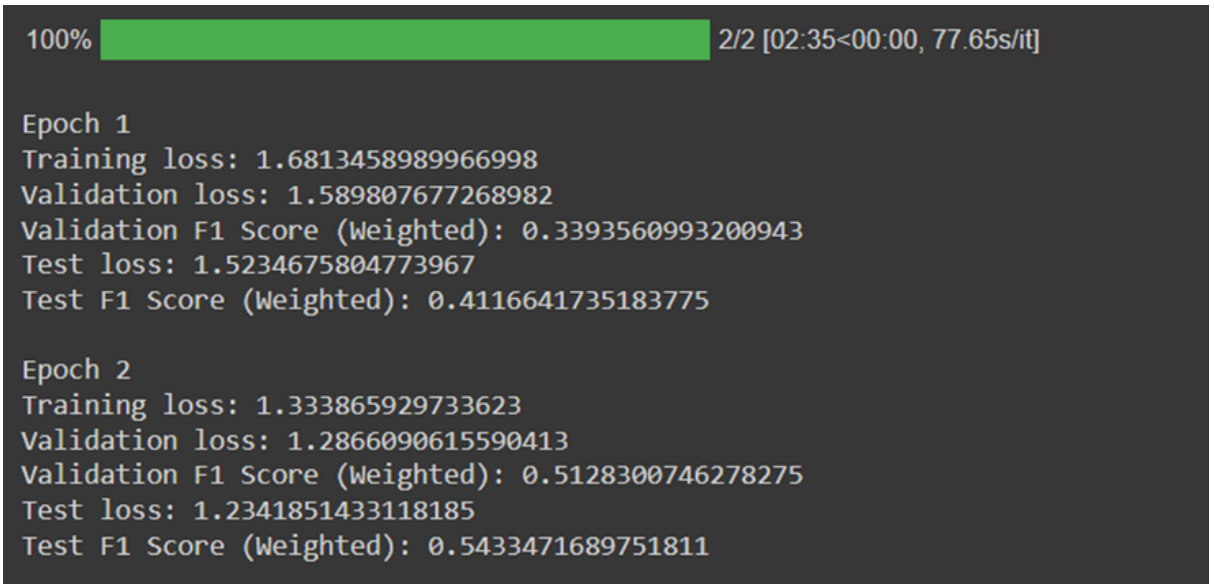
Figura 3: dataset com mil amostras sem usar a sumarização

```
100% ████████████████████████████████ 2/2 [02:38<00:00, 79.29s/it]

Epoch 1
Training loss: 1.6246982433579185
Validation loss: 1.4804683605829874
Validation F1 Score (Weighted): 0.39894136945075565
Test loss: 1.3795482079188028
Test F1 Score (Weighted): 0.45448223272554916

Epoch 2
Training loss: 1.205595002932982
Validation loss: 1.2386701544125875
Validation F1 Score (Weighted): 0.573778800206913
Test loss: 1.1644643823305765
Test F1 Score (Weighted): 0.6165892895788065
```

Figura 4: mesmo dataset da figura 3, porém utilizando a sumarização no dataset inteiro



Nesse segundo caso, podemos perceber um impacto maior ao utilizar a sumarização no dataset inteiro.

Considerando usar a sumarização no pré-processamento de textos e depois treinar um classificador, pode gerar uma perda de desempenho no classificador, mas que dependendo do problema pode ser aceitável. Por outro lado, a tarefa de sumarizar por si só já resolve diversos problemas no ambiente jurídico e podemos "validar" a sumarização medindo o quanto ela está impactando no classificador para julgar se é uma sumarização boa ou se o resultado não está sendo interessante.

Nesse link pode ser encontrado o notebook (colab) usado para fazer a sumarização dos textos: Sumarização - gate 08.ipynb

O notebook usado para treinar os modelos usando a arquitetura transformers pode ser encontrado nesse link: Gate8-14/12.ipynb

APÊNDICE 7

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 21 de dez. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Héber Júnior

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Conforme planejado na última semana, para essa entrega o foco foi trabalhar a parte da otimização do modelo de classificação de textos jurídicos. Para isso, a técnica usada foi a quantização. De uma forma simples, essa é uma técnica de compressão que visa reduzir o tamanho e o tempo de inferência dos modelos, com a perda mínima de desempenho. Nesse processo, os parâmetros (pesos) do modelo, que geralmente são representados como números de ponto flutuante de alta precisão (FP32) são convertidos para números de baixa precisão, como inteiros de 8 bits (INT8). Isso resulta em uma representação mais compacta do modelo, permitindo economizar espaço em memória e melhorar a eficiência computacional durante a inferência.

Para realizar essa etapa de quantização foi utilizado o framework Hugging Face que permite diversas formas de otimização através da biblioteca Optimum.

A quantização de fato trouxe melhorias para o modelo, reduzindo o tempo de inferência, o tamanho do modelo e com uma perda mínima de desempenho (F1-score).

	Modelo Original	Modelo quantizado
Tamanho (MB)	475	119
Tempo de inferência (s)	2864	2066
F1 - score	0.87	0.86

O desempenho do modelo foi pouco prejudicado e é aceitável considerando a redução no tamanho do modelo. Além disso, existem outras técnicas de otimização que podem reduzir ainda mais o tempo de inferência e serão trabalhadas na próxima entrega.

A descrição de tudo que foi feito pode ser encontrada no seguinte documento:

Gate 21/12/23

Planejamento: **[descrever o que pretende fazer para realizar a próxima ENTREGA]**

Pensando em otimizar ainda mais o modelo e explorar outras técnicas, para a próxima entrega vou adicionar o uso de otimização em grafo visando diminuir o tempo de inferência do modelo.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

[Documento citado no Termo de Aceite de Entrega de 21 de dezembro]

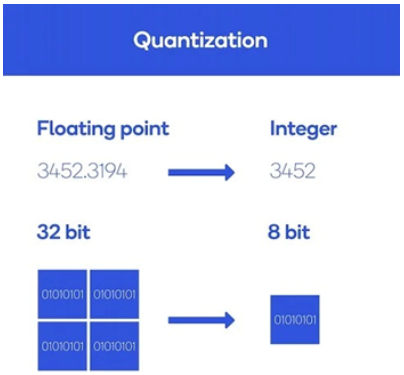
Otimização 1

Conforme apresentado nas últimas entregas, os modelos baseados na arquitetura transformers de fato entregaram melhores resultados na tarefa de classificação de textos jurídicos usando o dataset VICTOR (se comparado com as técnicas clássicas). Porém, esse ganho de desempenho na classificação não é de graça. São modelos mais robustos e por isso exigem mais recursos computacionais, como o uso de GPUs (Unidades de Processamento Gráfico) que oferecem o paralelismo para acelerar o treinamento, mas são mais caras.

Para lidar com esse problema de que os modelos transformers são mais pesados e caros computacionalmente, algumas técnicas podem ser usadas. Uma técnica muito utilizada é a técnica de quantização. No contexto desse trabalho, a quantização foi utilizada no pós-treino do modelo. Assim, após ter o modelo treinado, foram feitas otimizações visando reduzir o tempo de inferência. Um ponto importante é que o uso de quantização permite otimizar os modelos para realizar inferências usando CPUs e não necessitando de GPUs, que são mais caras.

A quantização de um modelo de classificação de textos é uma técnica de compressão que reduz o tamanho do modelo, mantendo um desempenho aceitável. Nesse processo, os parâmetros (pesos) do modelo, que geralmente são representados como números de ponto flutuante de alta precisão (FP32), são convertidos para números de baixa precisão, como inteiros de 8 bits (INT8). Isso resulta em uma representação mais compacta do modelo, permitindo economizar espaço em memória e melhorar a eficiência computacional durante a inferência.

Representação do processo de quantização (de FP 32 para INT8):



A biblioteca Optimum do Hugging Face permite diversas formas de otimização e ela foi utilizada nesses experimentos:

https://github.com/huggingface/optimum?tab=readme-ov-file

Algumas técnicas que essa biblioteca permite usar:

Features	ONNX Runtime
Graph optimization	✓
Post-training dynamic quantization	✓
Post-training static quantization	✓
Quantization Aware Training (QAT)	N/A
FP16 (half precision)	✓
Pruning	N/A
Knowledge Distillation	N/A

Abaixo segue a descrição do que foi feito:

O primeiro passo é já ter o modelo treinado. Após isso, deve-se seguir os seguintes passos:

- Converter o conjunto de dados (conjunto de teste) para o formato Dataset do Hugging Face.

- Carregamento do modelo treinado no formato ONNX (um formato otimizado que permite a integração com vários frameworks)

- Aplicação de quantização no modelo. Nessa etapa, foi feita a quantização dinâmica, o que significa que os parâmetros de quantização são calculados durante a execução do modelo. O modelo foi otimizado para fazer inferência em CPU e os pesos do modelo agora são representados como inteiro de 8 bits (INT8).

- O modelo permanece no formato ONNX após a quantização, garantindo compatibilidade com ONNX Runtime e outras estruturas compatíveis com ONNX.

Após ter o modelo quantizado e o modelo original, foram feitas comparações entre os dois:

Primeiro em relação ao tamanho, onde o modelo quantizado é cerca de 4 vezes menor que o modelo original:

```
Model file size: 475.77 MB
Quantized Model file size: 119.72 MB
```

Tabela de comparação dos dois modelos:

	Modelo Original	Modelo quantizado
Tamanho (MB)	475	119
Tempo de inferência (s)	2864	2066
F1 - score	0.87	0.86

Por fim, o código mede a latência de inferência dos modelos original e quantizado usando uma amostra de dados específica. O pipeline foi executado 300 vezes e são comparados o tempo do modelo original com o tempo do modelo quantizado.

Foi escolhida uma frase com cerca de 258 tokens. O “Vanilla model” representa o tempo de inferência do modelo original e o “Quantized model” representa o tempo do modelo

quantizado. Ao final é possível perceber que o modelo quantizado foi cerca de 1.43 vezes mais rápido que o modelo original.

```
Payload sequence length: 258
Vanilla model: P95 latency (ms) - 720.2803736998248; Average latency (ms) - 549.67 +\- 104.63;
Quantized model: P95 latency (ms) - 504.76368079971513; Average latency (ms) - 403.25 +\- 58.99;
Improvement through quantization: 1.43x
```

Além disso, outras técnicas de otimização serão abordadas na próxima entrega visando melhorar ainda mais o desempenho do modelo.

O notebook (colab) usado para realizar a quantização e os experimentos pode ser encontrado no link a seguir: Quantização - gate9.ipynb

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 11 de jan. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Héber Júnior

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Conforme planejado, nas entregas 9 e 10 o objetivo seria trabalhar com a otimização dos modelos de classificação de texto baseados na arquitetura Transformers com o intuito de reduzir o tempo de inferência.

Nessa entrega, fiz o fine tuning de um modelo BERT e também o fine tuning de uma versão distil do mesmo modelo para comparar o tempo de inferência deles. Além disso, também comparei esses modelos com a versão quantizada do BERT.

Após os experimentos, foi possível validar que o uso de modelos distil é uma boa estratégia, considerando que o tempo de inferência foi reduzido de forma significativa e que o desempenho (F1-score) sofreu uma perda insignificante.

O documento descrevendo todos os experimentos pode ser encontrado no link a seguir:

Gate 11/01/24

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Organizar os documentos para a elaboração e apresentação do TCC

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

Instituto de Informática (INF) - UFG
Alameda Palmeiras, Quadra D, Câmpus Samambaia
Fone: (62) 3521-1181 / Fax: (62) 3521-1182

[Documento citado no Termo de Aceite de Entrega de 11 de janeiro]

Otimização 2

Conforme meu planejamento, nas entregas 9 e 10 o objetivo seria trabalhar com formas para otimizar os modelos de classificação de textos jurídicos baseados na arquitetura transformers. Essas otimizações são focadas principalmente na parte de inferência dos modelos (mas também podem ser aplicadas durante o fine tuning) que muitas vezes é uma etapa muito custosa. Na entrega anterior, demonstrei que a quantização de fato é uma boa estratégia, considerando que o uso dessa técnica reduziu o tempo de inferência e diminuiu o tamanho do modelo, até mesmo facilitando para que a inferência pudesse ser feita em um CPU e não dependendo das GPUs.

Para essa entrega, a ideia inicial era trabalhar com a otimização em grafos, porém analisando mais profundamente como a biblioteca ONNX funciona (a biblioteca que utilizei para quantizar), percebi que implicitamente ao carregar os modelos com essa biblioteca já são feitas algumas otimizações em grafo. Por isso, optei por não adicionar mais uma etapa de otimização em grafo e partir para outra abordagem ainda pensando em otimização de modelos.

O foco para essa entrega é trabalhar com modelos do tipo Distil, mais especificamente o DistilBERT. O DistilBERT é uma versão destilada do BERT (menor, mais rápido, mais barato e mais leve). O treinamento do DistilBERT é realizado utilizando técnicas de destilação de conhecimento em grandes lotes, resultando em modelos mais leves que podem ser pré-treinados de maneira mais econômica.

A técnica de destilação, também conhecida como "knowledge distillation" (destilação de conhecimento), é uma abordagem na qual um modelo mais compacto, chamado de "aluno", é treinado para reproduzir o comportamento de um modelo maior e mais complexo, denominado "professor" ou "modelo de ensino".

No caso do DistilBERT, a destilação de conhecimento é utilizada durante a fase de pré-treinamento. O processo envolve treinar um modelo BERT maior como modelo de ensino (professor) e, em seguida, treinar um modelo DistilBERT menor como aluno para reproduzir as saídas do modelo BERT em um conjunto de dados específico. De uma forma simples, o objetivo da destilação é criar um modelo menor que possa imitar um modelo maior (um modelo menor com uma arquitetura semelhante ao modelo maior com um número reduzido de parâmetros).

Nessa entrega, resolvi fazer então a comparação dos 3 tipos de modelos: o modelo original, o modelo quantizado e o modelo versão distil. Para isso, usei como base o modelo BERT, que pode ser encontrado no Hugging face: https://huggingface.co/bert-base-uncased

A tabela abaixo mostra um comparativo entre os 3 modelos:

Modelo	Tamanho (Mb)	Tempo de inferencia (s)	F1-score
Bert	417	6684	0.75
Quantizado	105	4450	0.59
Distil	255	3319	0.73

Podemos perceber que de fato o uso de modelo Distil trouxe um ganho no tempo de inferência e uma perda mínima no F1-score, o que é aceitável.

Outro aspecto interessante de comentar também é que nem sempre o uso da quantização será bem sucedido. Nos experimentos da semana 9, utilizei a quantização e os resultados foram bastante satisfatórios, o tempo de inferência reduziu bastante e a perda de F1-score foi mínima. Já nesse caso, ao utilizar a quantização, o tempo de inferência também foi reduzido, porém o F1-score teve uma perda considerável, o que pode não ser aceitável em alguns cenários. Por isso, dependendo do modelo que estamos utilizando, o uso da quantização ou o uso de modelos distil pode se encaixar melhor. Como um especialista, é importante conhecer ambos os cenários e as possibilidades para que ao deparar com um problema desse tipo, saber qual a melhor estratégia a seguir.

O colab utilizado para fazer a comparação dos 3 modelos citados acima pode ser encontrado no link a seguir: Gate 10 - redução do tempo de inferência.ipynb

Referências:

[1] DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, https://arxiv.org/pdf/1910.01108.pdf