PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação *Lato Sensu* em Ciência de Dados e Big Data

Sandro Luiz de Aguilar

Classificação e sumarização de acórdãos de julgamento do Carf utilizando algoritmos de processamento de linguagem natural

Belo Horizonte
2021

Sandro Luiz de Aguilar

Classificação e sumarização de acórdãos de julgamento do Carf utilizando algoritmos de processamento de linguagem natural

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização em Ciência de Dados e Big Data como requisito parcial à obtenção do título de especialista.

Belo Horizonte

2021

SUMÁRIO

1. Introdução

1.1. Contextualização

O Contencioso Administrativo Federal é constituído pelas Delegacias de Julgamente (DRJ) e pelo Conselho Administrativo de Recursos Fiscais (CARF).

As DRJs, órgãos de deliberação interna e natureza colegiada, têm por finalidade julgar processos que versem sobre a aplicação da legislação referente aos tributos administrados pela Secretaria Especial da Receita Federal do Brasil do Ministério da Economia, conforme estabelecido em seu Regimento Interno.

Compete às DRJs apreciar, por decisão colegiada:

I - em primeira instância, a impugnação ou manifestação de inconformidade apresentada pelo sujeito passivo; e

II - em última instância, os recursos contra as decisões de que trata o inciso I do caput, em relação ao contencioso administrativo fiscal de pequeno valor, assim considerado aquele cujo lançamento fiscal ou controvérsia não supere sessenta salários mínimos. (Art. 2º da Portaria ME 340, de 2020)

O CARF, órgão colegiado, paritário, integrante da estrutura do Ministério da Fazenda, tem por finalidade julgar recursos de ofício e voluntário de decisão de 1ª (primeira) instância, exceto os recursos relativos ao contencioso administrativo fiscal de pequeno valor, bem como os recursos de natureza especial, que versem sobre a aplicação da legislação referente a tributos administrados pela Secretaria da Receita Federal do Brasil (RFB). (Art. 1º do Regimento Interno aprovado pela Portaria MF 343, de 2015)

Atualmente, há um estoque de processos gigantesco em apreciação nas DRJs e no Carf. Em torno de 250 mil nas DRJs e 120 mil no Carf.

1.2. O problema proposto

O julgamento de um processo é realizado a partir da análise dos seguintes documentos:

1 Termo de Verificação Fiscal e Auto de Infração: contêm o lançamento fiscal com os motivos, com a descrição dos fatos e com o enquadramento legal que embasaram a autuação.

2 Impugnação: contém as alegações do contribuinte feitas com o objetivo de afastar ou diminuir o lançamento fiscal.

O resultado do julgamento é escrito em um documento chamado Acórdão que contém dados descritivos do processo (número, interessado, assunto, nome dos julgadores, etc), um relatório com o resumo do lançamento e da impugnação e o voto que descreve a análise do julgador e o seu resultado.

O relatório de um acórdão requer um trabalho de sumarização pelo julgador dos documentos citados.

A sumarização automática com a utilização de processamento de linguagem natural é um importante recurso, nesse contexto, para acelerar o trabalho de julgamento de processos e, em conseqüência, auxiliar na diminuição do passivo existente.

Como a proposta da PUC para este TCC é ser um trabalho prático e a Receita Federal do Brasil espera a utilização na sua atividade, o ideal seria uma proposta de trabalho aplicada diretamente nos dados das Delegacias de Julgamento, pois é esta a minha unidade de trabalho e a minha atividade.

Entretanto, há uma impossibilidade legal de utilização dos documentos dos processos em discussão nas DRJ, em especial, os de interesse deste trabalho: os Termos de Verificação Fiscal, as impugnações e os acórdãos, por conterem informações de natureza econômico-fiscal protegidas pela regra de sigilo fiscal veiculada pelo art. 198 do Código Tributário Nacional, Lei 5.172, de 1966, acolhido pela Constituição Federal com força de Lei Complementar.

Diante disso, a alternativa encontrada foi a utilização dos acórdãos do Carf, pois esses são públicos, uma vez que o Carf não faz parte da Fazenda Pública que é a destinatária do comando legal supra.

Os acórdãos do Carf são formados por textos de conteúdo jurídico similares aos utilizados pelas DRJ, pois se tratam de uma segunda análise sobre os mesmos dados e sob um mesmo ordenamento jurídico, e são, portanto adequados ao objetivo deste trabalho.

Todos os processos julgados tem os seus dados descritivos e o inteiro teor do acórdão publicados no sítio do Carf, http://idg.carf.fazenda.gov.br/.

Os dados dos acórdãos serão analisados com o objetivo de serem classificados e sumarizados.

O objetivo principal do trabalho é avaliar as técnicas de sumarização automática de textos aplicadas a um corpus de textos jurídicos.

A estratégia adotada para isso será criar um modelo de classificação para ser testado com o conteúdo integral do voto e, também, com o conteúdo resumido.

Inicialmente, serão treinados diferentes modelos de classificação com base nos textos completos e tratados de diferentes maneiras. O modelo que apresentar os melhores resultados será escolhido para a avaliação dos resumos.

Um bom resumo permite que o destinatário avalie a compreensão do texto lido, incluindo a compreensão global, o desenvolvimento das idéias e a articulação entre elas (Machado, 2004).

No caso do resumo de um voto, é essencial que o leitor consiga distinguir o assunto, o resultado do voto e as controvérsias: motivos do lançamento, alegações da defesa e fundamentos da decisão.

A seleção de atributos (*feature selection)* é uma etapa fundamental em um projeto de Ciência de Dados e se utiliza de medidas estatísticas para estabelecer a correlação e dependência das variáveis preditoras com a variável alvo.

O desempenho de um modelo é fortemente influenciado pela qualidade e eficiência dos atributos.

Fazendo uma analogia com a classificação de textos, os atributos mais eficientes são as palavras que melhor se correlacionam com o assunto que se quer classificar.

O resumo de um texto irá diminuir a quantidade de palavras do texto original.

É premissa do trabalho que os melhores resumos são aqueles que conseguem extrair do texto original as frases que melhor representam os seus principais aspectos, as frases que melhor identificam o que é relevante.

Se assim o for, os melhores resumos conseguirão manter as palavras que melhor se correlacionam com a variável alvo permitindo que o modelo não tenha uma perda significativa de acurácia ao ser testado com o texto reduzido.

Para isso, os textos dos votos da base de teste serão sumarizados por meio de diversas técnicas de sumarização e, após a sumarização, serão submetidos à classificação pelo melhor modelo para obtermos a acurácia do modelo em classificar os textos sumarizados.

Pelo exposto, espera-se que os melhores sumarizadores sejam aqueles com a melhor acurácia na classificação, pois serão os que preservarão as características mais importantes do texto original.

Com essa estratégia, espera-se aferir automaticamente a qualidade dos resumos.

Entretanto, essa premissa pode não refletir a realidade.

Considerando que o trabalho tem a expectativa de subsidiar eventual criação de ferramenta para a criação de relatórios automáticos, é importante verificar se a classificação automática dos resumos está em consonância com a visão dos futuros usuários desses relatórios, os julgadores.

Para isso, elaborei uma pesquisa para conhecer e comparar a classificação dos resumos feita pelos julgadores.

A pesquisa e seus resultados serão mostrados ao final do trabalho.

2. Coleta de Dados

Todo o trabalho foi feito utilizando-se a linguagem de programação Python com a criação de scripts utilizando o framework ContÁgil[1] da Receita Federal.

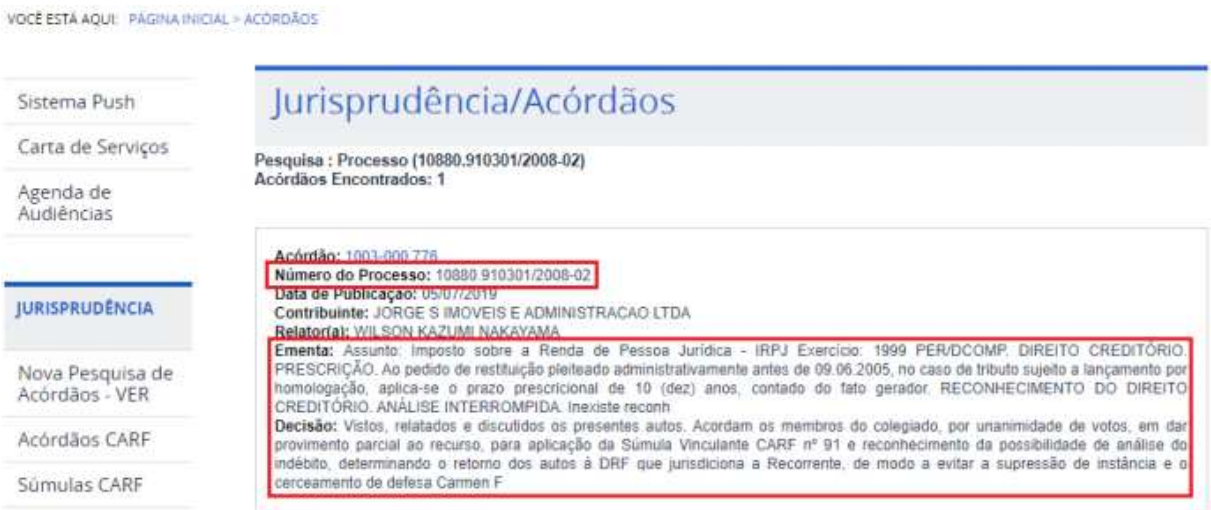
2.1 Script *TCC Big Data - Baixar acórdãos Carf*

A obtenção dos dados foi por meio da técnica de Web Scraping. Percorreu-se páginas do Carf[2] para obter dados descritivos do acórdão e a suas ementas gerando o dataset Carf – ementas com a seguinte estrutura.

Carf - ementas[3]

Nome da coluna/campo	Descrição	Tipo
Processo	Número do processo	Texto
Ementa	Síntese de uma decisão colegiada (acórdão)	Texto

Os dados das ementas foram obtidos na seguinte página[4].



Confira-se recorte do código da extração citada a seguir[5].

[1] O ContÁgil é um aplicativo desenvolvido na linguagem de programação JAVA que pode ser executado localmente, sem necessidade de estar conectado em rede, de uso interno da Receita Federal.
[2] https://carf.fazenda.gov.br/sincon/public/pages/ConsultarJurisprudencia/consultarJurisprudenciaCarf.jsf e https://carf.fazenda.gov.br/sincon/public/pages/ConsultarJurisprudencia/listaJurisprudenciaCarf.jsf
[3] Os dataframes obtidos por web scrapinp foram transformados em arquivos csv e gravados no Dive para utilização no Google Colab. O arquivo Carf - ementas foi gravado como content/drive/MyDrive/df_ementas.csv
[4] https://carf.fazenda.gov.br/sincon/public/pages/ConsultarJurisprudencia/consultarJurisprudenciaCarf.jsf

```
try:
    web.abrirPagina("https://carf.fazenda.gov.br/sincon/public/pages/ConsultarJurisprudencia/consultarJurisprudenciaCarf.jsf")
except Exception as erro:
    print(f'Processo {nrProcesso} com erro ao acessar página consultarJurisprudenciaCarf. \nErro: {erro}.')
    continue

try:
    # formulario para pesquisa
    form = web.getPaginaAtual().getFormulario("consultaJurisprudenciaForm")

    form.setCampo("valor_pesquisal", nrProcesso)
    form.setCampo("AJAXREQUEST", "_viewRoot")
    form.setCampo("consultaJurisprudenciaForm", "consultaJurisprudenciaForm")
    form.setCampo("j_id51", "j_id51")

    web.submeterFormulario(form)

    # formulario para detalhamento
    form = web.getPaginaAtual().getFormulario("formAcordaos")

    form.setCampo("tblJurisprudencia:0:numDecisao", "tblJurisprudencia:0:numDecisao")
    form.removeCampo("j_id89")
    form.removeCampo("botaoImprimir")

    web.submeterFormulario(form)

    tabela = web.getPaginaAtual().getTabela(0)

    for linha in range(0, tabela.getNumLinhas()):
        if str(tabela.getCelula(linha, "COLUNA-00"))[:6] == "Ementa":
            tab_A.setCelula(linhaCarf, "Processo", nrProcesso)
            tab_A.setCelula(linhaCarf, "Ementa", tabela.getCelula(linha, "COLUNA-0"))
            linhaCarf += 1

except Exception as erro:
    print(f'Processo {nrProcesso} com erro ao acessar o formulário consultaJurisprudenciaForm. \nErro: {erro}.')
    continue
```

De acordo com o Manual de redação e elaboração de atos administrativos da Secretaria da Receita Federal do Brasil, 2014, "*Ementa é a síntese da decisão, a que evidencia a regra aplicada ao caso concreto. Tem grande relevância para a compreensão da matéria objeto do processo e é citada como base na argumentação em outros julgamentos. Uma ementa bem elaborada facilita as pesquisas à jurisprudência administrativa*".

Este dataset será utilizado para a formação da lista de palavras bônus do sumarizador Edmundson. O procedimento será descrito no tópico relativo aos sumarizadores.

Em seguida, em outra página[6], em anexo, o pdf com o inteiro teor do acórdão foi baixado por meio do método exportaConteudoBinario[7] da classe WebExtrator[8] do ContÁgil.

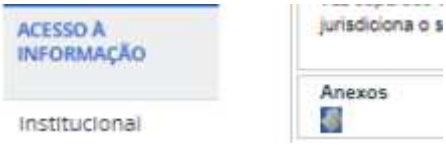
---

[5] Código integral do script disponível no github, link https://github.com/sla01/TCC_PUC_Minas

[6] https://carf.fazenda.gov.br/sincon/public/pages/ConsultarJurisprudencia/listaJurisprudenciaCarf.jsf

[7] Exporta o conteúdo binário que foi respondido na última requisição de página. Este é um método alternativo ao convencional getPaginaAtual. É recomendável para situações em que o resultado de uma consulta não é uma página HTML, mas sim o conteúdo de algum arquivo (como no caso de um arquivo PDF, por exemplo).

[8] Objeto que pode ser utilizado por uma linguagem de script para extrair dados de um serviço Web (HTTP). Este objeto está disponível para uma linguagem de script através do nome "web". Isto é, já está definido para qualquer linguagem de script um objeto do tipo WebExtrator cujo nome é "web", bastando executar seus métodos.



Confira-se recorte do código da extração citada a seguir[9].

```
try:
    # pegando o anexo
    form = web.getPaginaAtual().getFormulario("formAcordaos")
    form.removeCampo("formAcordaos:j_id64")
    form.removeCampo("botaoImprimir")
    form.setCampo("formAcordaos:_idcl","formAcordaos:j_id60:0:j_id61")
    web.submeterFormulario(form)
    web.exportaConteudoBinario("D:\\IACarf\\AcordaoBaixadoCarf\\"+nrProcesso+"_AcordaoCarf.pdf")
except Exception as erro:
    print(f'Processo {nrProcesso} com erro no download do acórdão. \nErro: {erro}.')
    continue
```

Foram baixados 18.594 arquivos pdf sendo 3.932 no dia 24/12/2020, de 11:34h às 20:21h, 2.386 no dia 17/01/2021, de 13:20h às 23:59h, 534 no dia 18/01/2021, de 00:00h às 01:54h, 4.947 no dia 18/01/2021, de 06:02h às 23:59h, no dia 19/01/2021, de 00:00h às 08:48h, 3.761 no dia 19/01/2021, de 11:19h às 15:49h.

2.2 Script *TCC Big Data - Ler pdf acordao e gravar dataframes com atributos e com voto em frases*

Para a leitura do pdf e obtenção dos textos do voto foi necessário excluir os textos com rotação e também foi preciso limitar a área de cobertura dos dados para excluir textos em segundo plano, marcas de cópia, número de folhas, conforme se pode observar na figura seguinte:

[9] Código integral do script disponível no github, link https://github.com/sla01/TCC_PUC_Minas

Os textos delimitados na figura ou sublinhados em vermelho são textos que prejudicam a legibilidade do documento por meios automáticos e não agregam nenhum valor ao sentido do texto.

A definição da exclusão foi feita desprezando-se os textos com rotação superior a 5 graus.

Quanto aos textos em segundo plano, foi executada uma leitura preliminar do texto para verificar a existência dos textos “Impresso em" “Assinado digitalmente em” na área superior a 782.30 no eixo y (função *parametro_PdfClasse_leitura_preliminar*). Existindo os textos, foi obtido o tamanho da fonte para posterior marcação como tamanho de fonte a desprezar na leitura definitiva.

Na leitura definitiva (função *parametro_PdfClasse_leitura_definitiva*), a área do pdf a ser lida foi delimitada entre 64.32 e 782.30 no eixo y.

Dessa forma, obteve-se o texto limpo dos pdf. Ressalte-se que essa estratégia de leitura foi definida após diversos testes de leitura em amostras dos arquivos. Confira-se o código a seguir.

```
def parametro_PdfClasse_leitura_preliminar(pdfClasse, pastaPesquisarPdf, Pdf):
    pdfClasse.setRotacaoMaxima(5)
    pdfClasse.setPosicaoYMinima(782.30)
    tamanhoFonteDesprezar = 0
    try:
        pdf = pdfClasse.abrirPDFComoTexto(pastaPesquisarPdf, Pdf)
        tab = pdfClasse.getDadosUltimaExtracao().getTamanhosAmostras(True)
        for linhaFonte in range(0, tab.getNumLinhas()):
            if 'Impresso em ' in tab.getCelula(linhaFonte, 'TEXTO') or 'Assinado digitalmente em'  in tab.getCelula(linhaFonte, 'TEXTO'):
                tamanhoFonteDesprezar = tab.getValorNumerico(linhaFonte, 'TAMANHO')
                break
    except:
        pass
    return tamanhoFonteDesprezar

def parametro_PdfClasse_leitura_definitiva(pdfClasse, tamanhoFonteDesprezar):
    pdfClasse.setRotacaoMaxima(5)
    pdfClasse.setPosicaoYMaxima(782.30)
    pdfClasse.setPosicaoYMinima(64.32)
    pdfClasse.addTamanhoDesprezar(tamanhoFonteDesprezar)
```

O resultado foi altamente satisfatório, pois em todos os pdf observados, o conteúdo do acordão estava completo e sem sujeiras.

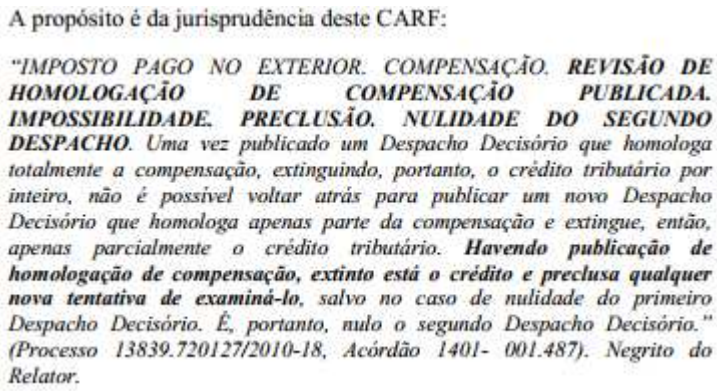
Após a recuperação do texto do acórdão, foi feita a localização dentro do pdf da parte relativa ao voto do julgador, pois o trabalho de classificação e sumarização seria feito com base no texto do voto.

Esse texto recebeu uma limpeza para prepará-lo para a tokenização em sentenças.

A limpeza do texto (função *limpar_texto*) excluiu o ponto separador de milhar, pois estava influenciando a tokenização; alterou a codificação de leis de 9.240/96 para 9240 de 96; substituiu o ponto em ementas/citações dentro do texto por vírgula, pois estavam sendo geradas sentenças com uma/duas palavras quando tratavam-se de títulos de ementas das citações; foram alteradas siglas/abreviaturas comuns para extenso, além de excluídos caracteres e palavras típicas utilizadas nos acórdãos, mas que caracterizavam verdadeiras *stop words* específicas destes textos (exemplo: dele conheço, assinado digitalmente, como voto, etc).

Por exemplo, citações no voto como abaixo geravam várias frases em virtude dos pontos finais após os títulos. Então, utilizando expressões regulares, transformei

os pontos em vírgulas para que os títulos das citações fossem interpretados como parte da próxima frase da citação.



A propósito é da jurisprudência deste CARF:

*"IMPOSTO PAGO NO EXTERIOR. COMPENSAÇÃO. **REVISÃO DE HOMOLOGAÇÃO DE COMPENSAÇÃO PUBLICADA. IMPOSSIBILIDADE. PRECLUSÃO. NULIDADE DO SEGUNDO DESPACHO**. Uma vez publicado um Despacho Decisório que homologa totalmente a compensação, extinguindo, portanto, o crédito tributário por inteiro, não é possível voltar atrás para publicar um novo Despacho Decisório que homologa apenas parte da compensação e extingue, então, apenas parcialmente o crédito tributário. **Havendo publicação de homologação de compensação, extinto está o crédito e preclusa qualquer nova tentativa de examiná-lo**, salvo no caso de nulidade do primeiro Despacho Decisório. É, portanto, nulo o segundo Despacho Decisório." (Processo 13839.720127/2010-18, Acórdão 1401- 001.487). Negrito do Relator.*

A tokenização desta citação ficou assim:

a propósito é da jurisprudência deste carf: imposto pago no exterior, compensação, revisão de homologação de compensação publicada, impossibilidade, preclusão, nulidade do segundo despacho, uma vez publicado um despacho decisório que homologa totalmente a compensação, extinguindo, portanto, o crédito tributário por inteiro, não é possível voltar atrás para publicar um novo despacho decisório que homologa apenas parte da compensação e extingue, então, apenas parcialmente o crédito tributário.;

havendo publicação de homologação de compensação, extinto está o crédito e preclusa qualquer nova tentativa de examiná-lo, salvo no caso de nulidade do primeiro despacho decisório.;

O código utilizou expressões regulares. Confira-se parte do código da função limpar_texto.

```
def limpar_texto(texto_recebido):
    texto_recebido = texto_recebido.replace('CPC.', 'CPC')
    texto_recebido = texto_recebido.replace('CC.', 'CC')
    texto_recebido = texto_recebido.replace('ART.','ARTIGO')
    texto_recebido = texto_recebido.replace('ARTS.','ARTIGOS')
    #tratamento valores numéricos em real. Exclui o ponto, pois estava influenciando a tokenização.
    texto_recebido = re.sub('([0-9]*)[\.]*([0-9]+,[0-9]+)', r'\1\2', texto_recebido)
    #altera a codificação de leis. De 9.240/96 para 9240 de 96 e nas ementas troca o ponto final para vírgula.
    texto_recebido = re.sub('([0-9]+)[.]+([0-9]+)\/([0-9]+)\.', r'\1\2 de \3,', texto_recebido)
    #altera a codificação de leis. De 9.240/96 para 9240 de 96
    texto_recebido = re.sub('([0-9]+)[.]+([0-9]+)\/([0-9]+)', r'\1\2 de \3', texto_recebido)
    #tratamento para ementas. substitui o ponto após duas palavras maiúsculas por vírgula.
    texto_recebido = re.sub('([A-Z0-9À-Ü]+[ ]+[A-Z0-9À-Ü]+)\.', r'\1,', texto_recebido)
    #tratamento para ementas. substitui o ponto após palavra maiúscula por vírgula.
    texto_recebido = re.sub('([A-ZÀ-Ü]+[0-9]*[A-ZÀ-Ü]*)\.', r'\1,', texto_recebido)
    #tratamento para ementas. substitui o ponto final por vírgula após palavras maiúsculas com número dentro de parênteses (ANO DE 2003).
    texto_recebido = re.sub('(\([A-ZÀ-Ü ]+[0-9]*\))\.', r'\1,', texto_recebido)
    #tratamento para ementas. substitui o ponto final por vírgula em datas. DE 1995.
    texto_recebido = re.sub('([A-ZÀ-Ü]+[ ]+[0-9]+)\.', r'\1,', texto_recebido)
```

Algumas limpezas foram determinadas após a observação do prejuízo causado pelos caracteres/*stop words* em amostras e pela minha experiência como julgador.

O texto do voto foi dividido em sentenças e foi gravado o dataset Carf_Voto_Dividido_Em_Frases com a seguinte estrutura.

Carf_Voto_Dividido_Em_Frases[10]

Nome da coluna/campo	Descrição	Tipo
Processo	Número do processo	Texto
Frase001 a 510	Voto dividido em frases	Texto

Veja a seguir recorte do dataframe em processamento no Colab.

	processo	frase001	frase002	frase003	frase004	frase005	frase006	frase007	frase008	frase009	frase010	frase011	frase012	frase013	frase014	frase015	frase016
0	10070000654200748	o recurso voluntário é tempestivo e reúne os r...	no que concerne à omissão de rendimentos, veri...	cabe mencionar nesse ponto que o valor de prev...	a folha 10, anexa comprovante de rendimentos p...	238,50 e desconto de r$ 1181,94 relativo à p...	não consta do referido comprovante desconto ...	assim o lançamento será alterado de ofício par...	do exame dos documentos fornecidos por esta f...	quanto à compensação indevida de irrf, não mer...	o sujeito passivo de fato não contestou o valo...	cumpre esclarecer que os recolhimentos efetuad...	por todo o exposto, voto por conhecer do recur...	()	NaN	NaN	NaN
1	10070000947200644	por ser tempestivo e por preencher as demais c...	conforme já descrito no relatório supra, o ple...	parágrafo único.	poderá ser deduzido, para fins de determinaç...	juntamente com o recurso voluntário, o contrib...	muito embora o alegado equívoco possa ter ocor...	nos termos do inciso xiii do artigo 224 do reg...	por outro lado, a leitura integrada dos artigo...	tal conclusão é corroborada pelo artigo 1 do a...	não obstante, em relação ao este valor sobre o...	por outro lado, igualmente comum é o fato de ...	contudo, o comprovante de rendimentos fornecid...	carlos alberto do amaral azeredo	NaN	NaN	NaN

O número de frases/sentenças foi limitado em 510. Processos que ultrapassaram esse limite foram desprezados.

Também foram desprezados 402 processos cujo pdf estava com erro de leitura por problema na ocerização ou qualquer outro problema.

Da leitura do pdf do acórdão foram extraídos os dados descritivos que formaram o dataset *Carf – atributos* com a seguinte estrutura.

Carf - atributos[11]

Nome da coluna/campo	Descrição	Tipo
Processo	Número do processo	Texto
Recurso	Tipo de recurso	Texto

[10] Os dataframes obtidos por web scrapinp foram transformados em arquivos csv e gravados no Dive para utilização no Google Colab. O arquivo Carf_Voto_Dividido_Em_Frases foi gravado como /content/drive/MyDrive/df_voto.csv

[11] Os dataframes obtidos por web scrapinp foram transformados em arquivos csv e gravados no Dive para utilização no Google Colab. O arquivo Carf – atributos foi gravado como /content/drive/MyDrive/df_atributos.csv

Materia	Tipo de matéria	Texto
Assunto	Tipo de assunto	Texto
Relator	Nome do relator	Texto
Presidente	Nome do presidente da turma	Texto
Conselheiros	Nome dos conselheiros que participaram do julgamento	Texto
Decisao	Quorum de votação da decisão	Texto
Voto vencedor	Se houve voto vencedor	Texto

Veja a seguir recorte do dataframe em processamento no Colab.

processo	recurso	materia	assunto	relator	presidente	conselheiros	decisao	voto vencedor
10070000654200748	voluntário	irpf - omissão de rendimentos	irpf	mônica renata mello fereira stoll	cláudia cristina noira passos da costa develly...	claudia cristina noira passos da costa devell...	unanimidade	NaN
10070000947200644	voluntário	irpf	irpf	determinando o recalculo do tributo devido nos...	carlos henrique de oliveira	carlos henrique de oliveira, ana cecília lust...	unanimidade	NaN

Para localização dos atributos no pdf foi utilizada pesquisa aos títulos dos atributos com expressões regulares, pois havia um padrão nos pdf. Confira-se a seguir a localização da matéria.

```
def atributo_materia(acordao):
    try:
        limites = "mat.ria(.*)\n"
        materia = re.search(limites, acordao).group(1)
        materia = materia.strip().lower()
    except:
        materia = ""
    return materia
```

O script utilizou as bibliotecas *os* (para percorrer a pasta com os arquivos pdf baixados), a biblioteca *shutil* (para mover de pasta os arquivos desprezados), a biblioteca *re* (para utilização na limpeza dos dados e na localização de textos no

pdf), as classes *PDFExtrator*[12] e *ArquivoPDF*[13] (para leitura e tratamento dos pdf) e a biblioteca *nltk* (para a tokenização das sentenças).

3. Processamento/Tratamento de Dados

Após a coleta dos dados pelo webscraping, todo o trabalho de análise, exploração dos dados, criação de modelos de Machine Learning, sumarização, geração de dados para pesquisa e apuração foram executados por meio de notebooks no Google Colab.

Todos os notebooks foram documentados no Colab em seções de texto utilizando a linguagem de marcação Markdown e anexados a este trabalho permitindo que a parte principal do texto não precise descrever códigos, pois estes estarão descritos com riqueza de detalhes em anexo.

O primeiro dos notebooks, TCC_p01_preparar_X_treinamento_e_y_teste, tem por objetivo preparar e salvar os dataframes X e y (treinamento e teste) que serão usados para treinar e testar os diferentes modelos (SpaCy e DCNN com stemer, lema e palavras originais) e, também, para a geração e classificação dos resumos. Como a base para as experiências precisa ser a mesma, após a preparação dos dados e a divisão em X e y, os dataframes serão salvos no drive. Assim, os dataframes poderão ser utilizados nos demais notebooks evitando, na medida do possível, procedimentos duplicados.

Foram importados os os datasets *Carf_Voto_Dividido_Em_Frases* (df_v) e *Carf – atributos (df_a)* utilizando a biblioteca *Pandas* e gerando 2 *dataframes*.

Criei o atributo *qtdfrase* no dataset *Carf_Voto_Dividido_Em_Frases* com a quantidade de frases/sentenças de cada processo. O quantitativo foi calculado por meio da função *soma_frases* e processado utilizando o método *apply* no eixo das linhas sendo computadas apenas as frases com pelo menos 4 palavras (*regex*).

---

[12] Objeto de scirpt que disponibiliza diversos recursos avançados para extrair textos a partir de arquivos PDF. Existem diversos métodos mais simples de converter arquivo PDF em texto, implementados em GerenciadorArquivos. Por exemplo, o método abrirPDFComoTexto. Porém, existem situações em que se quer utilizar alguns critérios mais avançados. Por exemplo, para considerar somente textos que apresentam uma determinada fonte, dado o nome da fonte. Esta classe se destina a realizar esses critérios avançados.

[13] Classe que disponibiliza diversas operações relativas a um arquivo PDF.

O objetivo de calcular o número de sentenças foi selecionar apenas os textos com pelo menos 10 sentenças, pois a sumarização de textos pequenos não é muito útil e poderia prejudicar a análise da qualidade dos sumarizadores.

Selecionei os 4 assuntos com maior quantidade de processos. Eram eles o Imposto de Renda da Pessoa Jurídica (IRPJ), o Imposto de Renda da Pessoa Física (IRPF), as Normas gerais de direito tributário (NGDT) e o Processo administrativo fiscal (PAF).

Foram selecionados: IRPF 3734; NGDT 2479; IRPJ 2331 e PAF 1927.

Os assuntos são definidos e cadastrados pelo relator do voto. IRPJ são os acórdãos relativos à controvérsia do tributo Imposto de Renda da Pessoa Jurídica. IRPF são os relativos à Imposto de Renda de Pessoa Física. NGDT não são relativos a um tipo de tributo, mas às normas gerais que permeiam todo o sistema tributário como, por exemplo, o instituto da Decadência. Por fim, PAF, também é matéria que permeia o sistema sendo as normas processuais relativas ao processo administrativo fiscal.

Criei o atributo *voto* no dataset *Carf_Voto_Dividido_Em_Frases* com a concatenação de todas as frases/sentenças. A concatenação foi feita por meio da função *concatena_frases* também utilizando o método *apply* no eixo das linhas.

A quantidade de frases era variável e as frases sem conteúdos foram geradas como NaN pelo *Pandas*. As frases Nan foram desprezadas.

O *dataframe* de treinamento e validação foi criado a partir dos *dataframes* df_v e df_a com os atributos voto e assunto e o atributo processo foi a chave de ligação entre os *dataframes*.

O *dataframe* foi dividido em treinamento e validação (teste) na proporção de 80 e 20%. Para isso, utilizei o método *train_test_split* da biblioteca *sklearn*.

O atributo alvo assunto foi transformado em um dicionário com as 4 categorias e o dado booleano *True* ou *False*. (tabela verdade)

4. Análise e Exploração dos Dados

Nessa seção você deve mostrar como foi realizada a análise e exploração dos seus. Mostre as hipóteses levantadas durante essa etapa e os padrões e insights identificados.

Descrever a etapa 2 do notebook 01.

2. unção preprocessamento (essa função será replicada nos notebooks de classificação e sumarização).

- A feature voto contém o documento inteiro do voto, ou seja, uma coleção de frases e palavras com pontuações, valores numéricos, stop words, etc. A função do preprocessamento faz a limpeza nesse texto devolvendo apenas as palavras que serão usadas nas classificações. A função está descrita em docstring reproduzido a seguir.
    - Faz uma limpeza do texto recebido excluíndo pontuações, stop words, números em moedas, leis, etc, letras soltas, palavras com duas letras, hífens iniciais, r$, espaços duplos e símbolos.
    - O objetivo é deixar apenas palavras que serão a base para as classificações.
    - :parâmetro texto: recebe o texto integral de um voto.
    - :parâmetro tipo: escolher entre 'lema', 'stemer' e 'original'. Lema faz lematização das palavras, stemer faz a stemização e original mantém as palavras sem alterações.
    - :retorno str_lista: string com a lista das palavras do voto após a limpeza dos dados.

4. Preprocessar o voto passando como parâmetro lema, stemer e original. Dessa forma, serão obtidos 3 dataframes, cada um utilizando uma forma de tratamento das palavras. Os dataframes serão salvos no drive para posterior utilização no treinamento dos modelos.
5. Salvar no Drive os dataframes gerados pelo preprocessamento.

- Utilizei o encoding Windows-1252, uma codificação de caracteres de byte único do alfabeto latino, usada por padrão nos componentes herdados do Microsoft Windows que se mostrou adequada aos textos utilizados. Os arquivos gravados no drive foram os seguintes:
    - X_treinamento_lema.csv
    - X_treinamento_stemer.csv
    - X_treinamento_original.csv

5. Criação de Modelos de Machine Learning

5.1 Classificação dos votos

A primeira parte do trabalho trata da classificação dos votos por assunto para escolha do modelo de *machine learning* que será usado para a classificação dos resumos.

Foram testados dois modelos para a classificação. O primeiro modelo utilizou a biblioteca SpaCy[14] e o segundo utilizou o TensorFlow[15].

[14] SpaCy é uma biblioteca de código aberto gratuita para Processamento de Linguagem Natural (PNL) avançado em Python. Ela foi desenvolvida pela Explosion AI e é mantida por Matthew Honnibal e Ines Montani. Pode ser usada para construir sistemas de extração de informações ou de compreensão de linguagem natural, ou para pré-processar texto para aprendizado profundo. A biblioteca é escrita na linguagem Cython, que é a extensão C

Para a categorização pela SpaCy foi usada a arquitetura padrão que é o conjunto empilhado de um modelo linear de saco de palavras e um modelo de rede neural. A rede neural é construída sobre uma camada Tok2Vec e usa atenção[16].

O categorizador de texto prevê categorias em um documento inteiro não sendo necessário transformar as palavras em vetores. Isso é feito internamente pela biblioteca. O modelo pode aprender um ou mais rótulos, e os rótulos são mutuamente exclusivos - há exatamente um rótulo verdadeiro por documento.

A variável alvo é passada como um dicionário com as categorias existentes.

Esse procedimento está descrito passo a passo no notebook TCC_p02_Classificacao_Spacy anexo ao final do TCC. Lá, estão os códigos python utilizados com descrição da seqüência de eventos e com comentários nos códigos.

Ver onde colocar: Foi utilizado 50 épocas, mas a estabilização do erro indicou que as 30 épocas utilizadas na lematização e nas palavras originais era suficiente. Com 50 épocas o tempo de processamento foi excessivo sem melhora nos resultados. Por essa razão, mantive o treinamento dos demais com 30 épocas.

Para a categorização com o TensorFlow, optei por utilizar uma rede neural convolucional.

Uma rede neural convolucional profunda (DCNN) consiste em muitas camadas de redes neurais. Dois tipos diferentes de camadas, convolucionais e pooling, são normalmente alternados.

Nas camadas de convolução, a informação passa por vários filtros (que na prática são matrizes numéricas) com a função de acentuar padrões regulares locais, ao mesmo tempo em que vão reduzindo a dimensão dos dados originais. Os resultados de vários filtros são sumarizados por operações de pooling. Na parte mais profunda das convoluções, espera-se que os dados num espaço dimensional reduzido contenham informação suficiente sobre esses padrões locais para atribuir

---

do Python. Suporta quase 30 idiomas, fornece fácil integração de aprendizagem profunda e promete robustez e alta precisão. As informações sobre Spacy foram obtidas no sítio oficial https://spacy.io/.

[15] O TensorFlow é uma plataforma completa de código aberto para machine learning. Ele tem um ecossistema abrangente e flexível de ferramentas, bibliotecas e recursos da comunidade que permite aos pesquisadores levar adiante machine learning de última geração e aos desenvolvedores criar e implantar aplicativos com tecnologia de machine learning. https://www.tensorflow.org/?hl=pt-br

[16] https://spacy.io/api/textcategorizer e https://spacy.io/api/architectures#TextCatEnsemble

um valor semântico ao dado original. Esses dados passam então por uma estrutura de FFN clássica para a tarefa de classificação[17].

Avaliar figuras no PowerPoint

No caso da DCNN, é necessário transformar o documento em um saco de palavras e as palavras em vetores numéricos. Os rótulos também são transformados em categorias numéricas.

Em ambos os modelos de classificação, foram testadas três formas de tratamento das palavras do voto:

1) Stemização[18]: é o processo de reduzir palavras flexionadas (ou às vezes derivadas) ao seu tronco (stem), base ou raiz, geralmente uma forma da palavra escrita.

2) Lematização[19]: No contexto puramente lexicográfico a palavra dicionarizada recebe a denominação de lema ou forma canônica. A lematização é o ato de representar as palavras através do infinitivo dos verbos e masculino singular dos substantivos e adjetivos.

3) Original: nenhum tratamento foi realizado nas palavras.

A seguir, recortes do dataframe X_treinamento nos três formatos descritos[20].

	Palavras originais	voto
0	recurso tempestivo cumpre requisitos legais admissibilidade razão tomo conhecimento passo apreciar recurso tempestivo cumpre requisitos legais admissibilidade razão tomo conhecimento passo aprecia...	
1	registrada oportunidade caso veio julgamento recurso voluntário tempestivo recorrente devidamente representada preliminar nulidade recorrente alega despacho decisório homologou dcomp nulo fundamen...	
2	recurso voluntário apresentado atende pressupostos admissibilidade sendo digno conhecimento surge presente lide respeito necessidade expressa declaração homologação compensação tela compulsando au...	
3	julgamento processo segue sistemática recursos repetitivos regulamentada artigo anexo ricarf aprovado portaria junho presente litígio aplica-se decidido acórdão proferido julgamento processo parad...	
4	julgamento processo segue sistemática recursos repetitivos regulamentada artigo anexo ricarf aprovado portaria junho presente litígio aplica-se decidido acórdão proferida julgamento processo parad...	

[17] https://iaexpert.academy/2020/06/08/os-tipos-de-redes-neurais/
[18] https://pt.wikipedia.org/wiki/Stemização
[19] http://www.nilc.icmc.usp.br/nilc/download/lematizacao_versus_steming.pdf
[20] Dataframes gravados no drive pelo notebook TCC_p01_preparar_X_treinamento_e_y_teste descrito em anexo

Lematização: infinitivo dos verbos e masculino singular.
Exemplos: tomo=tomar, cumpre=cumprir, expressa=expresso, respeito=respeitar

	voto
0	recurso tempestivo cumprir requisito legar admissibilidade razão tomar conhecimento passar apreciar recurso tempestivo cumprir requisito legar admissibilidade razão tomar conhecimento passar aprec...
1	registrar oportunidade casar vir julgamento recurso voluntário tempestivo recorrente devidamente representar preliminar nulidade recorrente alegar despachar decisório homologar dcomp nulo fundamen...
2	recurso voluntário apresentar ater pressuposto admissibilidade ser dignar conhecimento surgir presentar lidar respeitar necessidade expresso declaração homologação compensação tela compulsar auto ...
3	julgamento processar seguir sistemático recurso repetitivo regulamentar artigo anexar ricarf aprovar portar junho presentar litigio aplica-se decidir acórdão proferir julgamento processar paradigm...
4	julgamento processar seguir sistemático recurso repetitivo regulamentar artigo anexar ricarf aprovar portar junho presentar litigio aplica-se decidir acórdão proferir julgamento processar paradigm...

Stemização: reduzir palavras flexionadas (ou às vezes derivadas) ao seu tronco (stem), base ou raiz.
Exemplos: tomo=tom, cumpre=cumpr, expressa=express, respeito=respeit

	voto
0	recurs tempes cumpr requisit legal admissibil ra tom conhec pass apreci recurs tempes cumpr requisit legal admissibil ra tom conhec pass apreci recorr apresent dcomp ra pag irpj códig darf compens...
1	registr oportun cas vei julg recurs voluntári tempes recorr devid represent prelimin nulidad recorr aleg despach decisóri homolog dcomp nul fundament legal anális darf dipj dctf verifica-s real re...
2	recurs voluntári apresent atend pressupost admissibil send dign conhec surg pres lid respeit necess express declar homolog compens tel compuls aut verifica-s contribuint ped manifest inconform det...
3	julg process seg sistemá recurs repeti regulament artig anex ricarf aprov port junh pres litígi aplica-s decid acórd profer julg process paradigm pres process fic vincul transcreve-s soluç litígi ...
4	julg process seg sistemá recurs repeti regulament artig anex ricarf aprov port junh pres litígi aplica-s decid acórd profer julg process paradigm pres process fic vincul process paradigm analis po...

Como se pode perceber, o método de tratamento das palavras reflete diretamente na quantidade de vocábulos do corpus. Isso porque palavras originais diferentes podem ter a mesma raiz ou o mesmo lema. Ao aplicar o tratamento, há uma redução da dimensionalidade. Sem nenhum tratamento, utilizando as palavras em seu formato original, o total de vocábulos foi 53.448. Quando as palavras foram reduzidas ao lema, o total caiu para 34.140. Quando a redução foi pela raiz, o total foi 27.107[21].

Feita a preparação dos dados, treinei o modelo SpaCy e o modelo DCNN utilizando a stemização, a lematização e as palavras originais.

Todos os 6 modelos treinados foram salvos no Drive para posterior utilização na classificação dos dados sumarizados. Apenas o modelo com a melhor acurácia será testado com os dados sumarizados.

Todos os treinamentos foram feitos com base no mesmo conjunto de documentos (X e y treinamento) e foram validados no mesmo conjunto de teste (X e y teste).

Após os testes, o modelo DCNN utilizando palavras lematizadas apresentou a maior acurácia e foi o escolhido para a fase de teste dos dados sumarizados.

[21] Dados obtidos no notebook TCC_p01_preparar_X_treinamento_e_y_teste descrito em anexo.

5.2 Classificação dos votos sumarizados

Sumarização de Textos com Processamento de Linguagem Natural – IA Expert

A área de Processamento de Linguagem Natural – PLN (Natural Language Processing – NLP) é uma subárea da Inteligência Artificial que tem como objetivo tornar os computadores capazes de entender a linguagem humana, tanto escrita quanto falada. Alguns exemplo de aplicações práticas são: tradutores entre idiomas, tradução de texto para fala ou fala para texto, chatbots, sistemas automáticos de perguntas e respostas, geração automática de descrições para imagens, adição de legendas em vídeos, classificação de sentimentos em frases, dentre várias outras! Outro exemplo importante de aplicação é a sumarização automática de documentos, que consiste em gerar resumos de textos. Vamos supor que você precise ler um artigo com 50 páginas, porém, não possui tempo suficiente para ler o texto integral. Nesse caso, você pode utilizar um algoritmo de sumarização para gerar um resumo deste artigo. O tamanho deste resumo pode ser configurável, ou seja, você pode transformar 50 páginas em um texto com somente 20 páginas que contenha somente os pontos mais importantes do texto!

Nesta fase, utilizei 7 sumarizadores da biblioteca Sumy e duas técnicas de sumarização codificadas em python para resumir os dados de teste.

Usando a codificação em python das técnicas:

Algorítimo de Luhn: Lunh propôs a criação automática de resumos de texto selecionando-se as frases mais importantes do texto de acordo com a sua freqüência. As palavras significativas são aquelas que aparecem com mais frequência no texto, mas, ao mesmo tempo, não fazem parte das stop words[22] do

[22] Na computação, uma palavra vazia é uma palavra que é removida antes ou após o processamento de um texto em linguagem natural. São palavras que podem ser consideradas irrelevantes para o conjunto de resultados a ser exibido. Não existe uma lista universal de palavras vazias usadas por todas as ferramentas de processamento de linguagem natural e nem todas ferramentas fazem uso de uma lista dessas palavras. Qualquer grupo de palavras pode ser escolhido como grupo de "palavras vazias" de acordo com o objetivo do processamento. https://pt.wikipedia.org/wiki/Palavra_vazia

idioma. As palavras com freqüências muito altas ou muito baixas seriam desconsideradas[23].

Similaridade do cosseno: Neste método, a importância das frases é dada pela nota de semelhança de cada frase com as demais calculada pela similaridade do cosseno[24] e classificada com a utilização do algoritmo PageRank[25].

Usando os sumarizadores da biblioteca Sumy[26]

LuhnSummarizer: Utiliza o algoritmo de Luhn supra.

EdmundsonSummarizer: Método heurístico com pesquisa estatística anterior - o aprimoramento do método Luhn mencionado anteriormente. Edmundson adicionou mais 3 heurísticas ao método para medir a importância das sentenças. Ele encontra as chamadas palavras pragmáticas, as palavras que estão nos cabeçalhos e a posição dos termos extraídos. Portanto, este método possui 4 sub-métodos e a combinação adequada deles resulta no método de Edmundson. Importante ressaltar que este método é o mais dependente do idioma porque precisa da lista de palavras bônus, palavras estigmatizadas e palavras irrelevantes (chamadas de palavras nulas).

Por outro lado, a construção da lista de palavras permite uma adequação maior ao seu conjunto de dados.

Neste trabalho, a criação da lista de palavras bônus foi elaborada a partir do dataset Carf – ementas. E pelas últimas frases.

[23] Fabrício Shigueru Catae – Classificação Automática de Texto por meio de similaridade de palavras: um algoritmo mais eficiente, 2012.
[24] Similaridade de cosseno é uma medida de similaridade entre dois vetores diferentes de zero de um espaço de produto interno. É definida como igual ao cosseno do ângulo entre eles, que também é o mesmo que o produto interno dos mesmos vetores normalizados para ambos terem comprimento 1. https://en.wikipedia.org/wiki/Cosine_similarity
[25] é um algoritmo utilizado pela ferramenta de busca Google para posicionar websites entre os resultados de suas buscas. O PageRank mede a importância de uma página contabilizando a quantidade e qualidade de links apontando para ela. Não é o único algoritmo utilizado pelo Google para classificar páginas da internet, mas é o primeiro utilizado pela companhia e o mais conhecido.
[26] A descrição das técnicas de sumarização foi obtida em https://miso-belica.github.io/sumy/summarizators.html

Conforme já descrito, as ementas são uma síntese da decisão e trazem uma sequência de palavras chaves que indicam o assunto, a tese jurídica e o conteúdo da decisão. Logo, são bastante adequadas a formação de um corpus que influencie o sumarizador na escolha das melhores sentenças.

A lista das palavras irrelevantes foi formada pelas stop words da SpaCy e da NLTK adicionadas dos nomes dos relatores. CONFIRMAR

Não houve formação de lista de palavras estigmatizadas.

LsaSummarizer: Método algébrico - o método mais avançado é independente do idioma, mas também o mais complicado (computacionalmente e mentalmente). O método é capaz de identificar sinônimos no texto e os tópicos que não estão explicitamente escritos no documento. Usa semântica latente e avaliação resumida.

LexRankSummarizer: Abordagem não supervisionada inspirada nos algoritmos PageRank e HITS - algoritmos criados para a rede mundial de computadores. Eles tentam encontrar conexões entre as frases e identificar aquelas relacionadas com as palavras / tópicos mais significativos.

TextRankSummarizer[27]: é um modelo de classificação baseado em gráfico. A maneira de decidir a importância de um vértice dentro do gráfico é pela “votação” ou “recomendação”. Quando um vértice se vincula a outro, ele basicamente está lançando um voto para aquele outro vértice. A conexão entre duas sentenças é determinada se houver uma relação de “similaridade” entre elas, onde a “similaridade” é medida em função de sua sobreposição de conteúdo. Quanto maior o número de votos que são lançados para um vértice, maior a importância do vértice. Abordagem não supervisionada inspirada nos algoritmos PageRank.

SumBasicSummarizer: Método frequentemente usado como linha de base na literatura para comparar a pontuação dos algoritmos. Não tem nenhuma vantagem especial sobre o LSA ou o TextRank.

RandomSummarizer: Método de teste que não deve ser usado para aplicações do mundo real. É usado apenas durante a avaliação dos resumos para comparação com os outros algoritmos. A ideia por trás disso é que se algum

[27] https://web.eecs.umich.edu/~mihalcea/papers/mihalcea.emnlp04.pdf

sumarizador tiver uma pontuação pior do que este, provavelmente é um algoritmo muito ruim ou há alguma falha/bug sério na implementação.

Após a sumarização dos dados de testes, eles foram submetidos ao modelo de classificação selecionado, DCNN utilizando palavras lematizadas, e as acurácias de cada sumarizador foram comparadas elegendo-se os melhores sumarizadores com base na acurácia.

5.3 Pesquisa:

Na perspectiva da Receita Federal, o objetivo dos trabalhos de TCC no curso de pós-graduação oferecido aos seus servidores é a possibilidade de aplicação às suas atividades. Sob esta ótica, é importante validar a qualidade dos sumarizadores sob o olhar de servidores que poderiam beneficiar-se das suas conclusões.

Embora para o TCC, isso não seja pré-requisito, decidi elaborar uma pesquisa sobre os sumarizadores com a participação de julgadores.

Como a participação na pesquisa é voluntária e para evitar a falta de colaboradores, optei por submeter ao crivo dos colegas apenas 3 sumarizadores.

Selecionei o sumarizador Edmundson que obteve os melhores resultados na avaliação automática proposta neste trabalho.

Selecionei o sumarizador Lsa que obteve o segundo melhor resultado e é considerado um método avançado capaz de identificar sinônimos no texto e tópicos que não estão explicitamente escritos no documento, conforme descrição supra.

Não selecionei o Luhn – Sumy considerando que o Edmundson é o aprimoramento do Luhn com o acréscimos das palavras bônus, estigmatizadas e irrelevantes. Sendo assim, preferi selecionar outra técnica.

Por fim, selecionei o sumarizador TextRank que é baseado em grafos e é uma abordagem não supervisionada inspirada nos algoritmos PageRank. O TextRank obteve a terceira melhor nota de avaliação,

Em seguida, selecionei 48 processos aleatoriamente. Todos foram sumarizados pelos 3 sumarizadores. Cada julgador recebeu os 48 processos sendo 16 sumarizados por cada um dos 3 sumarizadores. Assim, todo julgador poderia avaliar resumos elaborados pelos 3 sumarizadores.

Para melhor distribuir a pesquisa, os julgadores nascidos nos dias 01 a 10 receberam um pdf com os 16 primeiros processos sumarizados por Edmundson, os 16 seguintes por TextRank e os 16 finais por Lsa. Para os nascidos entre os dias 11 e 20, a ordem foi Lsa, Edmundson e TextRank. Para os demais, a ordem foi TextRank, Lsa e Edmundson.

O arquivo continha o Resumo do voto e o Voto completo.

A avaliação proposta foi a seguinte:

1. Responder qual é o assunto do processo dentre os 4 possíveis: IRPJ, IRPF, NGDT e PAF.
2. Responder o resultado do julgamento (improcedente ou procedente/procedente em parte).
3. Dar uma nota para a compreensão do texto. Se considera que conseguiu compreender o voto resumido (lide, argumentos, decisão). Nota de 1 a 10.
4. Dar uma nota para a coesão do texto, para o encadeamento das frases. Como o resumo exclui frases, se aparentemente as frases mais importantes foram mantidas. Se não parece haver uma lacuna entre as frases. Nota de 1 a 10.

Com o objetivo de não onerar os colaboradores e incentivar a participação, selecionei processos com número de frases entre 11 e 12. Foram selecionados 12 processos de cada uma das classes de classificação.

Esse procedimento foi executado pelo notebook TCC_p08_Gerar_Pdf_Sumarizado descrito em anexo.

Por fim, considerando a participação voluntária, os julgadores foram orientados a contribuir com a quantidade de avaliações possíveis. No caso de não avaliarem todos os resumos, que o fizessem em ordem aleatória para permitir a avaliação de diferentes modelos.

É importante ressaltar que todas as definições da pesquisa, infelizmente, seguiram a minha intuição e não critérios estatísticos que deveriam nortear uma boa pesquisa, mas que não foram aplicados por impossibilidades de conhecimento e disponibilidade de tempo.

6. Apresentação dos Resultados

Nessa seção você deve apresentar os resultados obtidos. Apresente gráficos, *dashboards*, conte a sua história de forma bastante criativa. Aqui você pode utilizar os modelos de Canvas propostos por Dourard (clique aqui) ou por Vasandani (clique aqui).

6.1 Modelo de Canvas

Classificação e sumarização de acórdãos de julgamento do Carf utilizando algoritmos de processamento de linguagem natural		
Definição do problema	Resultados e previsões	Aquisição de dados
Avaliar a qualidade de sumarizadores para a utilização no julgamento de processos administrativos com o objetivo de facilitar e acelerar o julgamento como enfrentamento do estoque crescente e diante da insuficiência de pessoas.	Pretende-se classificar um texto com conteúdo jurídico tributário por assunto. O preditor são as palavras contidas em um voto do Carf e a variável alvo é o assunto. Pretende-se classificar resumos de textos com conteúdo jurídico tributário por assunto. O preditor são as palavras contidas nos resumos e a variável alvo é o assunto.	Os dados são públicos e estão disponíveis no sítio do Carf. Há necessidade de construir Web Scraping para a raspagem dos dados necessários ao trabalho.
Modelagem	Avaliação do modelo	Preparação de dados
Para a fase de classificação, os modelos apropriados para o trabalho são aqueles que permitem a classificação com base em processamento de linguagem natural. Serão utilizados modelos da biblioteca SpaCy e do TensorFlow com rede neural convolucional. Para a fase de sumarização, principal foco do trabalho, é necessária a apli-	Os modelos serão avaliados pela quantidade de classificações corretas, ou seja, pela acurácia. O melhor modelo será aplicado aos textos sumarizados. O melhor sumarizado será aquele que permitir ao modelo a melhor acurácia.	A preparação dos dados consiste, sinteticamente, em transformar os votos em um conjunto de palavras. Nesse processo, pontuações, palavras sem conteúdo, números, etc, precisam ser excluídos.

cação de técnicas de sumarização de textos.		

6.2 Treinamento da SpaCy

O modelo da SpaCy foi treinado recebendo o conjunto de palavras do voto lematizadas, stemizadas e sem tratamento, ou seja, palavras originais.

Os únicos parâmetros passados para o modelo foram o número de épocas e a quantidade de registros no treinamento para a atualização dos pesos (minibatch).

Após a execução dos treinamentos, foram plotados gráficos de progressão dos erros em relação às épocas (o número de execução por época foi de 14 – razão entre a quantidade de registros 7.124 e o número de lotes).

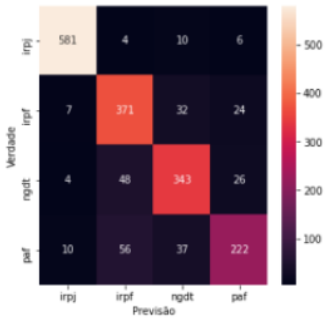
Foram feitos testes utilizando 30 e 50 épocas. Com 50 épocas, o tempo de processamento foi excessivo e sem melhora nos resultados.

Conforme demonstram os gráficos seguintes, em todos os casos, há uma estabilização dos erros após 200 batches (aproximadamente 14 épocas) indicando que o número de épocas usado no treinamento, 30, estava satisfatório.

Após o treinamento, os modelos foram aplicados aos dados de teste e os resultados obtidos mostrados em gráficos de matriz de confusão com mapa de calor.

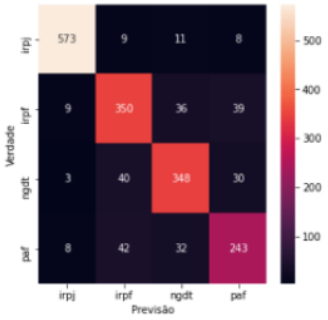
Lematização

Acurácia 85,17%



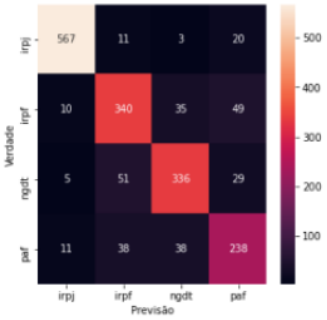
Stemização

Acurácia 85,00%



Palavras originais

Acurácia 83,15%



Em relação à acurácia, o modelo que usou as palavras lematizadas apresentou o melhor resultado com 85,17% de acertos. O segundo melhor modelo foi o que usou palavras stemizadas. Esse modelo teve índice de acerto de 85,00%, ou seja, apresentou uma acurácia muito próxima ao modelo lematizado.

Já o modelo que não fez nenhuma transformação nas palavras usando-as em seu formato original apresentou a menor acurácia com 2% de perda em relação aos demais modelos.

6.2 Treinamento com Rede Neural Convolucional Profunda utilizando o TensorFlow

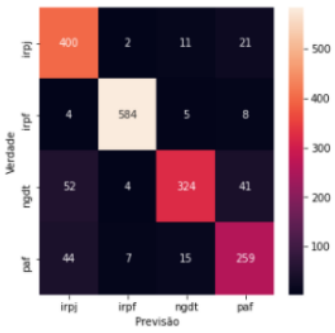
Para o treinamento da DCNN, os parâmetros passados foram: dimensões de embeding = 200, número de filtros por região da camada convolucional = 100, número de neurônios na camada escondida = 256, número de épocas 5, taxa de dropout = 0,2 e número de classes = 4.

Como o foco do trabalho é a comparação entre os sumarizadores e não a qualidade dos classificadores, não foi feita otimização dos parâmetros.

Após o treinamento, os modelos foram aplicados aos dados de teste e os resultados obtidos mostrados em gráficos de matriz de confusão com mapa de calor.

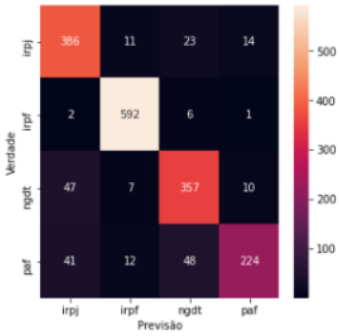
Lematização

Acurácia 87,98%



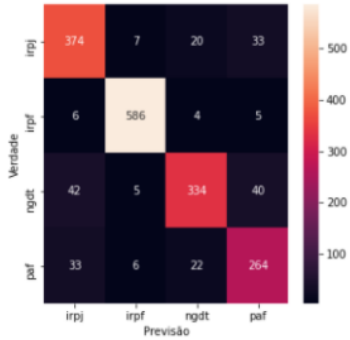
Stemização

Acurácia 87,53%



Palavras originais

Acurácia 87,47%



Percebe-se que a classificação por acurácia manteve a mesma ordem da SpaCy com palavras lematizadas, stemizadas e sem tratamento. Entretanto, a diferença entre os modelos foi pequena não apresentando a mesma queda no caso de palavras sem tratamento.

6.3 Modelo de melhor acurácia aplicado aos sumarizadores

A acurácia dos modelos Spacy situou-se na casa de 85% e os da DCNN, na casa de 87% de acurácia.

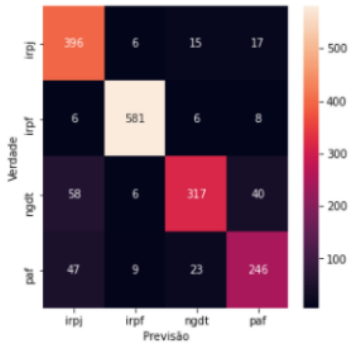
Observa-se em ambos os casos que, embora não destacadamente, os modelos trabalhados com o corpus de palavras lematizadas obtiveram os melhores resultados e os piores resultados foram para classificações sem tratamento das palavras.

O melhor resultado foi obtido pela DCNN com palavras lematizadas.

Portanto, foi com este modelo que os dados de teste sumarizados foram classificados. Os resultados foram os seguintes:

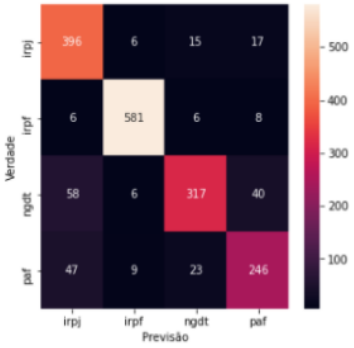
Luhn - Sumy

Acurácia 86,46%



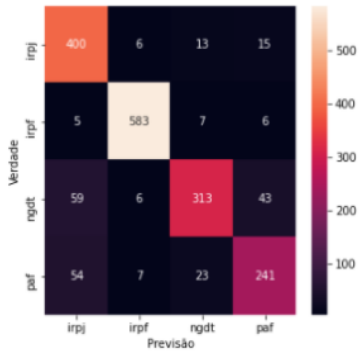
Lsa

Acurácia 86,46%



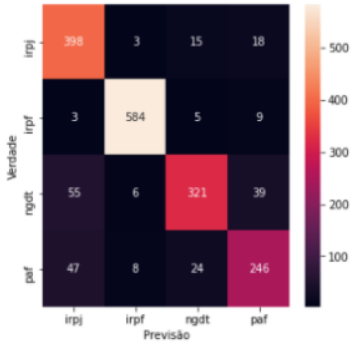
TextRank

Acurácia 86,29%



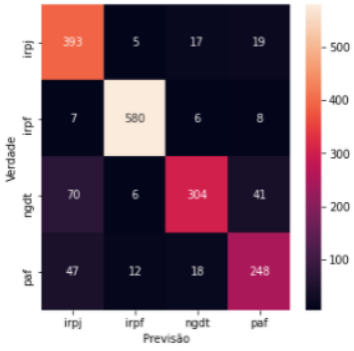
Edmundson

Acurácia 86,97%



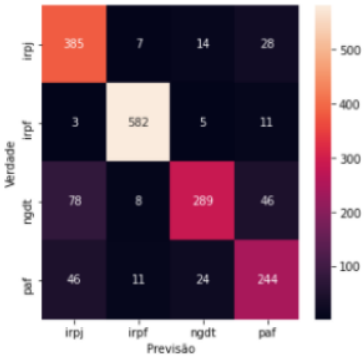
LexRank

Acurácia 85,62%



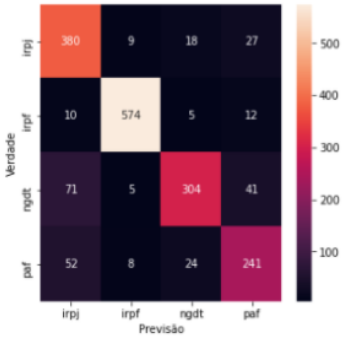
SumBasic

Acurácia 84,22%



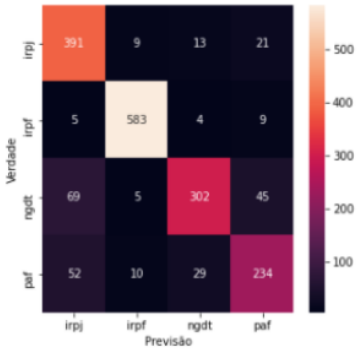
Random

Acurácia 84,16%



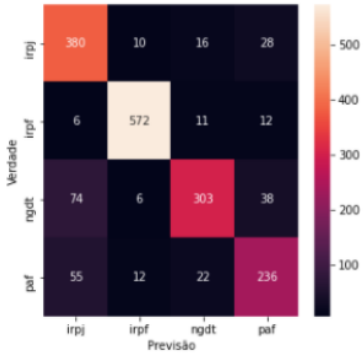
Similaridade Cosseno

Acurácia 84,78%



Luhn - codificado

Acurácia 83,71%



A seguir, a relação dos sumarizadores classificados pela acurácia.

Sumarizador	**Acurácia**

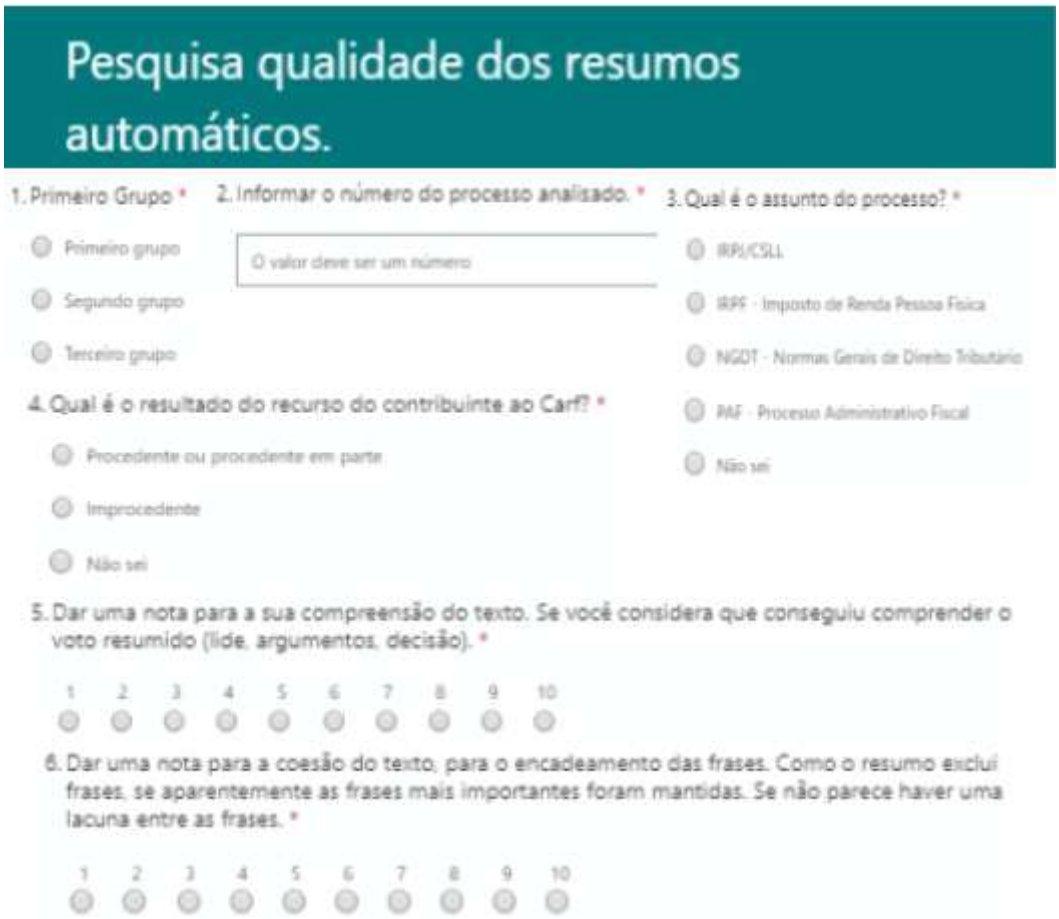
Edmundson	86,97%
Luhn - Sumy	86,46%
Lsa	86,46%
TextRank	86,29%
LexRank	85,62%
Similaridade Cosseno	84,78%
SumBasic	84,22%
Random	84,16%
Luhn - codificado	83,71%

Como se pode perceber, houve uma queda na acurácia comparando-se à classificação com o texto completo. Entretanto, considerando-se que os resumos foram elaborados com 40% do texto completo, a perda de acurácia pode ser considerada muito pequena.

Se a premissa do trabalho está correta, a qualidade dos sumarizadores é satisfatória, pois conseguiram manter as palavras relevantes para a classificação.

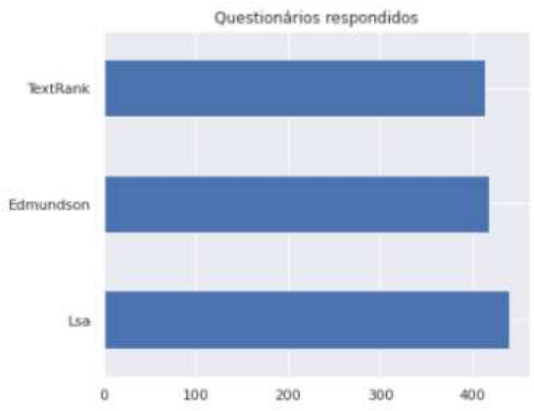
6.4 Avaliação de sumarizadores por humanos

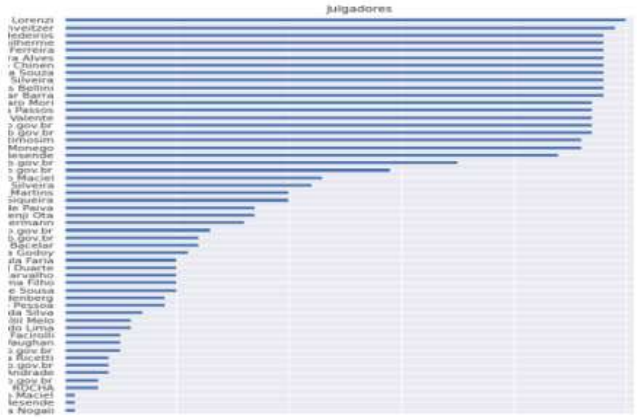
A pesquisa foi realizada por meio de um formulário no plataforma colaborativa Microsoft Teams disponível na Receita Federal e ficou disponível para os julgadores no período de 23/03/2021 até 17/04/2021 quando foi feita a apuração dos dados disponíveis.



Os julgadores deveriam responder qual era o assunto e o resultado do recurso, dar nota de compreensão e de coesão, para, no conceito de Machado (2004) supra, indicar a qualidade do resumo sob a ótica dos julgadores.

Foram respondidos 1.272 questionários por 53 julgadores das Delegacias de Julgamento da Receita Federal, conforme mostram os gráficos a seguir:



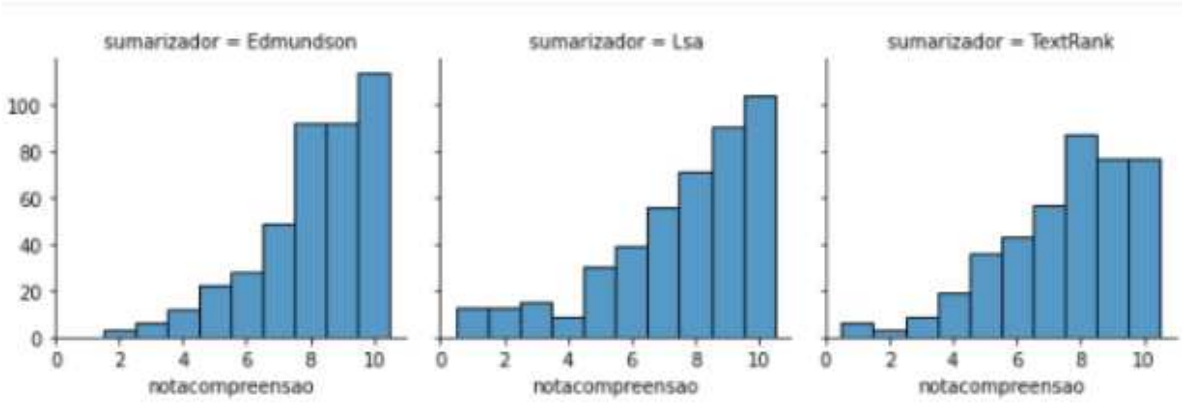


O número de questionários respondidos por tipo de sumarizador ficou bastante equilibrado.

Os julgadores responderam os questionários de acordo com a disponibilidade, não havendo a obrigatoriedade de responder todos, por isso a diferença quantitativa entre eles.

A seguir, gráficos relativos às notas de compreensão e coesão.

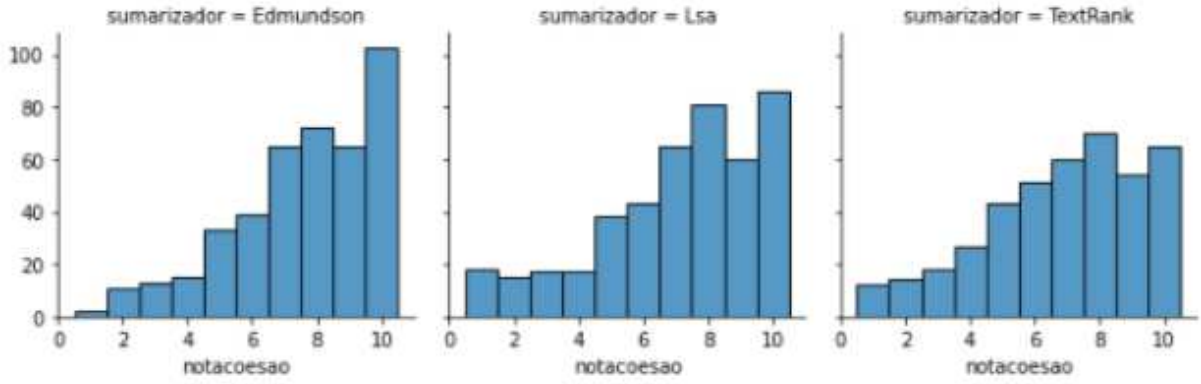
Notas de compreensão por sumarizador - histograma



O resultado da avaliação pelos julgadores da compreensão dos textos ficou coerente com a avaliação automática. Aqui, também, a ordem foi Edmundson, Lsa e TextRank.

A visualização dos gráficos mostra que embora a nota média tenha sido alta, existem algumas notas muito baixas indicando que o sumarizador não conseguiu entregar um resumo satisfatório. Entretanto, a maioria dos resumos obteve notas satisfatórias.

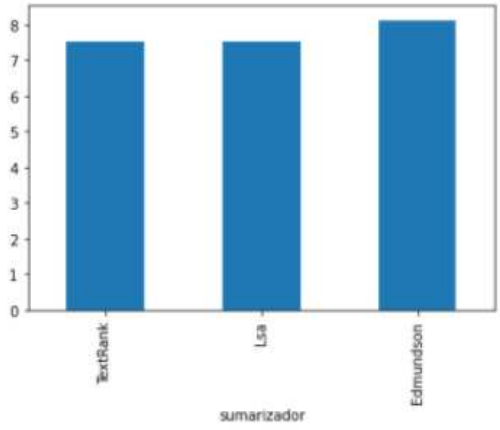
Notas de coesão por sumarizador - histograma



As notas de coesão mantêm a distribuição majoritariamente positiva, mas há uma queda previsível. É que os sumarizadores são do tipo extrativo e é natural que frases de conexão com a utilização de preposições, conjunções e advérbios, palavras vazias de conteúdo na lógica do processamento de linguagem natural sejam as mais fortes candidatas à exclusão.

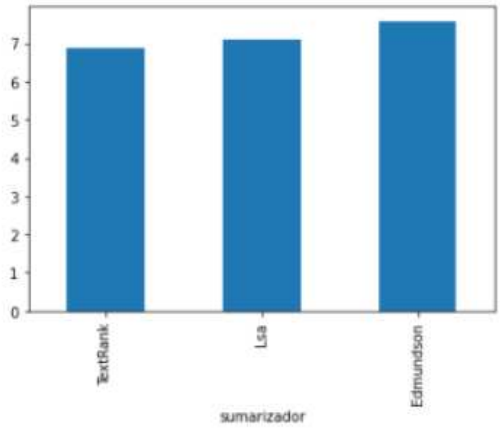
Nota de compreensão:

Edmundson 8,12, Lsa 7,53 e TextRank 7,51



Nota de coesão:

Edmundson 7,57, Lsa 7,08 e TextRank 6,85

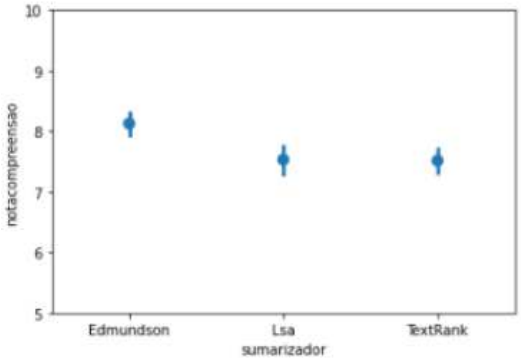


As notas médias de compreensão e de coesão indicam que os sumarizadores geraram resumos com bom nível e que podem ser úteis na atividade de julgamento.

A seguir, os gráficos de pontos mostram estimativas da tendência central para as notas de compreensão e coesão pela posição dos pontos do gráfico de dispersão e fornecem indicação da incerteza em torno dessa estimativa usando barras de erro.

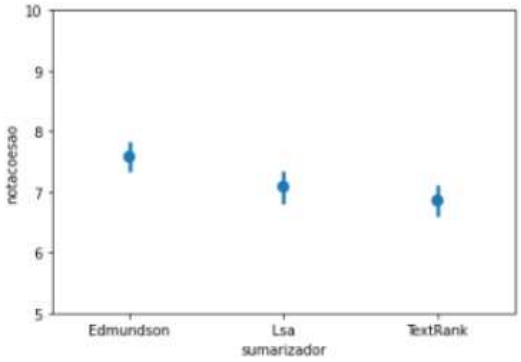
Nota de compreensão:

Edmundson 8,12, Lsa 7,53 e TextRank 7,51

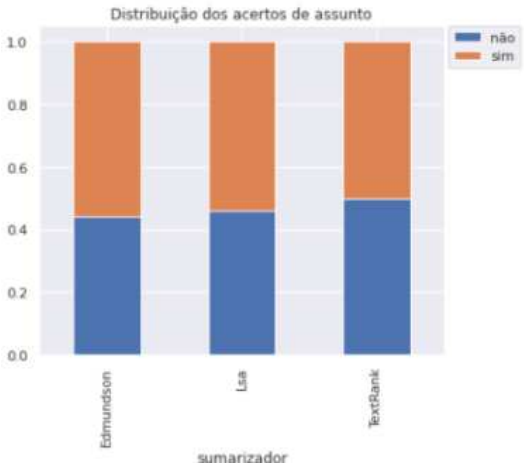


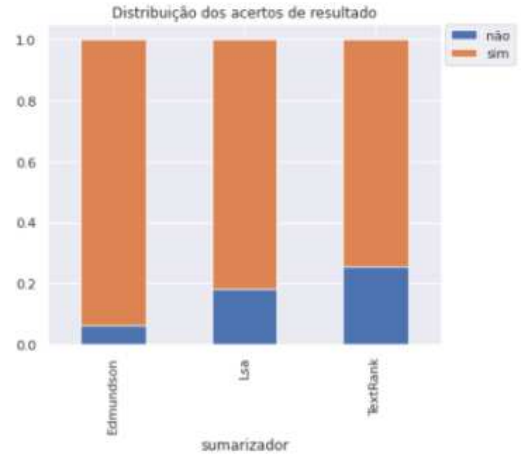
Nota de coesão:

Edmundson 7,57, Lsa 7,08 e TextRank 6,85



A seguir, o resultado da classificação pelos julgadores dos resumos por assunto e por resultado de julgamento.

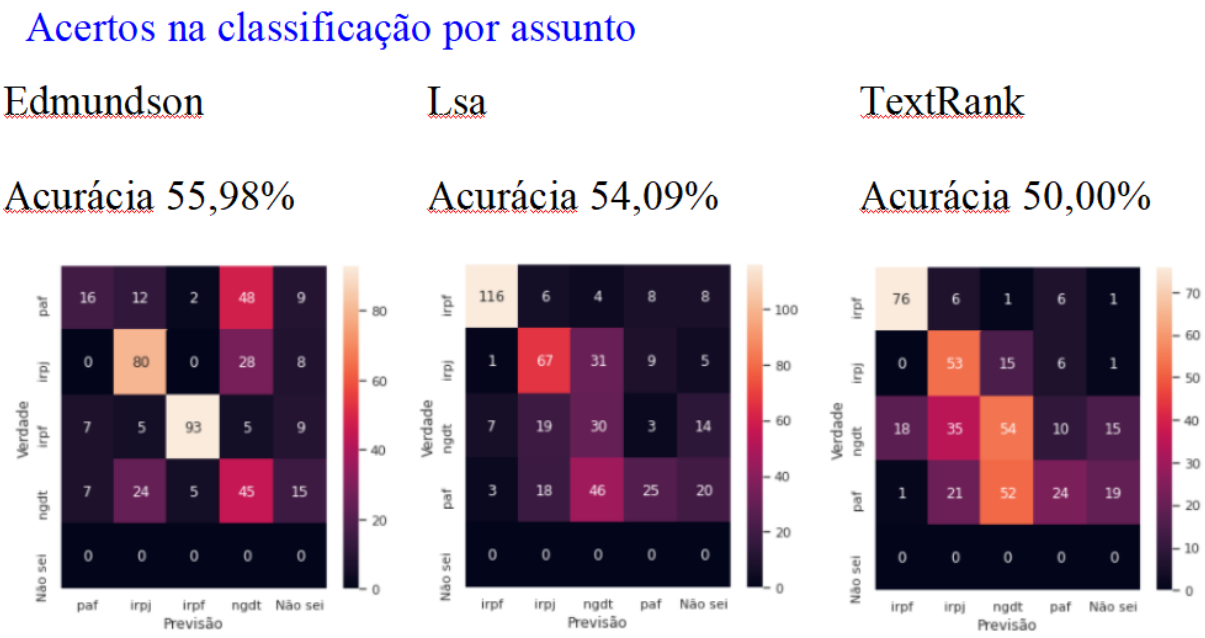




Percebe-se que o índice de acerto na classificação dos textos por assunto caiu muito na avaliação por humanos. A classificação feita pela rede neural convolucional acertou 86,97% com Edmundson, 86,46% com Lsa e 86,29% com TextRank. Já os julgadores acertaram 55,98% com Edmundson, 54,09% com Lsa e 50,00% com TextRank.

Esses resultados foram surpreendentes, em princípio, mas podem ser explicados pelo grau de especialização dos julgadores. Há julgadores que julgam apenas processos de Pessoa Física, outros de IRPJ/CSLL, e outros que julgam matérias que sequer estavam presentes neste trabalho como PIS/Cofins, IPI, Comércio Exterior. Os processos de Normas Gerais de Direito Tributário e Processo Administrativo Fiscal permeiam todo tipo de processo.

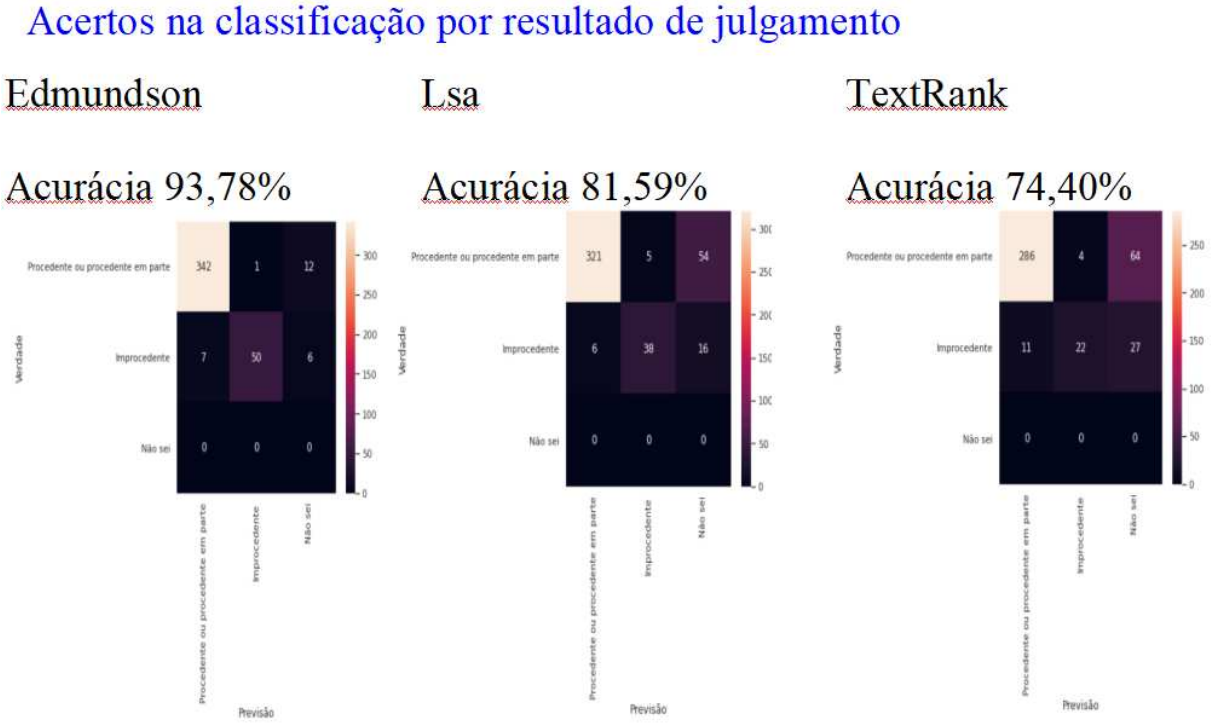
Nesta lógica, seria mais esperado que um modelo treinado em todos os tipos de matérias tivesse um desempenho melhor que um modelo especialista. É possível que os julgadores tenham melhores resultados se testados nas suas especialidades como um modelo que se adaptou à sua base de treinamento. Isso é um bom exemplo de *overfitting* em humanos.




```
Percentual de acertos de assunto para os julgadores que leram o resumo Edmundson: em 418 tentativas, acerto de 234, ou 55.98%
Percentual de acertos de assunto para os julgadores que leram o resumo Lsa: em 440 tentativas, acerto de 238, ou 54.09%
Percentual de acertos de assunto para os julgadores que leram o resumo TextRank: em 414 tentativas, acerto de 207, ou 50.00%
```

Quando se trata de indicar o resultado de julgamento, temos uma pergunta neutra em relação à especialização. O desconhecimento da matéria não prejudica a percepção sobre o resultado.

Neste quesito, os resultados foram muito bons indicando que os sumarizadores mantiveram as frases responsáveis por este importante conteúdo para a compreensão global deste tipo de texto.




```
Percentual de acertos de resultado para os julgadores que leram o resumo Edmundson: em 418 tentativas, acerto de 392, ou 93.78%
Percentual de acertos de resultado para os julgadores que leram o resumo Lsa: em 440 tentativas, acerto de 359, ou 81.59%
Percentual de acertos de resultado para os julgadores que leram o resumo TextRank: em 414 tentativas, acerto de 308, ou 74.40%
```

Conclusão

O objetivo do trabalho era avaliar a qualidade de sumarizadores com base na acurácia de classificação utilizando um modelo de Machine Learning.

Para isso, foram comparados os modelos de classificação da biblioteca SpaCy com uma Rede Neural Convolucional – DCNN utilizando o TensorFlow.

Em ambos os casos, os modelos foram treinados em um corpus de palavras originais, lematizadas e stemizadas.

O modelo que apresentou o melhor resultado foi a Rede Neural Convolucional utilizando palavras lematizadas.

Foram testados sete sumarizadores da biblioteca Sumy e codificados outros dois sumarizadores. Os textos sumarizadores foram utilizados para a classificação pelo modelo DCNN e os sumarizadores com melhor desempenho foram considerados mais qualificados.

Para validar a conclusão do trabalho, foi feita uma pesquisa com julgadores avaliando a qualidade de três dos sumarizadores testados.

O resultado da pesquisa foi coerente com o resultado da avaliação automática na ordem de classificação dos sumarizadores e a nota de compreensão e coesão dada pelos julgadores indicou a possibilidade de aproveitamento deste trabalho para a elaboração de ferramenta para a geração de relatórios resumos na atividade de julgamento na Receita Federal.

Com base nos resultados, vejo com muito otimismo essa possibilidade, mas também com precaução.

A nota de coesão menor que a nota de compreensão indica a necessidade de tratamento dos resumos pelos julgadores.

Nessa linha, entendo que a alternativa mais viável é a criação de uma ferramenta que proponha um resumo para ser validado e ajustado pelo usuário.

Dessa forma, haverá o ganho da utilização dos sumarizadores, mas ponderado com a experiência do usuário.

7. Links

Aqui você deve disponibilizar os links para o vídeo com sua apresentação de 5 minutos e para o repositório contendo os dados utilizados no projeto, scripts criados, etc.

Link para o vídeo: youtube.com/...

Link para o repositório: https://github.com/sla01/TCC_PUC_Minas

8. Figuras

Figura 1 - acórdão

DF CARF MF Fl. 131

S3-TE03
Fl. 131

MINISTÉRIO DA FAZENDA
CONSELHO ADMINISTRATIVO DE RECURSOS FISCAIS
PRIMEIRA SEÇÃO DE JULGAMENTO

Processo nº	10805.908224/2011-11
Recurso nº	Voluntário
Acórdão nº	1803-002.610 – 3ª Turma Especial
Sessão de	24 de março de 2015
Matéria	PERD/COMP
Recorrente	LAB HORN - Laboratório Especializado em dosagens hormonais Ltda
Recorrida	FAZENDA NACIONAL

ACÓRDÃO GERADO NO PGD-CARF PROCESSO 10805.908224/2011-11

ASSUNTO: IMPOSTO SOBRE A RENDA DE PESSOA JURÍDICA - IRPJ

Exercício: 2004

LUCRO PRESUMIDO. PERCENTUAIS. REQUISITOS ESPECÍCOS. PROVA. INTERPRETAÇÃO DA LEGISLAÇÃO TRIBUTÁRIA. POSICIONAMENTO JUDICIAL SUJEITO À SISTEMÁTICA DOS RECURSOS REPETITIVOS. VINCULAÇÃO DA ESFERA ADMINISTRATIVA.

1. Os percentuais de lucro presumido, no imposto sobre a renda e na contribuição social sobre o lucro líquido, definidos para serviços equiparados à hospitalares, para exercícios anteriores à 2009, independem de comprovação de requisitos específicos, limitado a exigência do objeto próprio da atividade.

2. Possibilidade de reconhecimento de crédito pleiteado, se o conjunto probatório e as condições especiais da demanda justifiquem a relativização do formalismo processual, com base no princípio da verdade real.

Vistos, relatados e discutidos os presentes autos.

Acordam os membros do Colegiado, por unanimidade de votos, pelo provimento do recurso voluntário, com reconhecimento do direito creditório.

(assinado digitalmente)

Carmen Ferreira Saraiva – Redatora Designada Ad Hoc e Presidente

Composição do colegiado. Participaram do presente julgamento os Conselheiros: Sérgio Rodrigues Mendes, Roberto Armond Ferreira da Silva, Meigan Sack Rodrigues, Ricardo Diefenthaeler, Fernando Ferreira Castellani e Carmen Ferreira Saraiva.

1

REFERÊNCIAS

Um projeto/relatório técnico de Ciência de Dados não requer revisão bibliográfica. Portanto, a inclusão das referências não é obrigatória. No entanto, caso você deseje incluir referências relacionadas às tecnologias ou às metodologias usadas em seu trabalho, relacione-as de acordo com o modelo a seguir.

SOBRENOME DO AUTOR, Nome do autor. **Título do livro ou artigo.** Cidade: Editora, ano.

MACHADO, Anna Rachei Machado, Eliane Gouvêa Lousada, Lilia Santos Abreu-Tardelli. **Resumo**. São Paulo:Parábola Editorial, 2004.

SOBRENOME DO AUTOR, Nome do autor. **Título do livro ou artigo.** Cidade: Editora, ano.

SOBRENOME DO AUTOR, Nome do autor. **Título do livro ou artigo.** Cidade: Editora, ano.

SOBRENOME DO AUTOR, Nome do autor. **Título do livro ou artigo.** Cidade: Editora, ano.

SOBRENOME DO AUTOR, Nome do autor. **Título do livro ou artigo.** Cidade: Editora, ano.

SOBRENOME DO AUTOR, Nome do autor. **Título do livro ou artigo.** Cidade: Editora, ano.

APÊNDICE

Programação/Scripts

Cole aqui seus scripts em Python e/ou R.

Gráficos

Cole aqui workflows (KNIME, RapidMiner, etc), gráficos e figuras que você tenha gerado e não colocou no texto principal.

Tabelas

Cole aqui tabelas de dados que você tenha gerado e não colocou no texto principal.