Pontifícia Universidade Católica
do Rio de Janeiro

Projeto Final de Programação

Extrator de Precedentes para Decisões do Supremo Tribunal Federal e Superior Tribunal de Justiça

José Luiz Nunes

Orientadora: Profa. Simone Diniz Junqueira Barbosa

12 de Julho de 2020

Conteúdo

1 Introdução

Precedentes vêm recebendo maior atenção em alterações legislativas no Brasil. Essas modificações tiveram como objetivo aumentar sua importância no processo decisório judicial no país. Isto foi necessário porque, ao contrário de sistemas de *common law*, como nos Estados Unidos, Inglaterra e Índia, onde precedentes judiciais são o centro da tomada de decisão judicial, vinculando futuros casos, o sistema de *civil law* brasileiro revolvia na interpretação de normas positivadas. Neste cenário, mesmo decisões da mais alta Corte do país, o Supremo Tribunal Federal, não vinculavam as decisões de juízes de primeira instância.

Este cenário começa a mudar com a aprovação da Emenda Constitucional 45 em 2004, e sua regulamentação pela Lei Lei nº 11.418/06. Com essas alterações surgem as Súmulas Vinculantes, que vinculam todo o judiciário, assim como a Administração Pública, e o instituto de repercussão geral. Além disso, o Código de Processo Civil de 2015 (Lei 13.105 de 2015) ampliou ainda mais a importância de precedentes, ao ampliar o leque de decisões que vinculam o tribunal e órgãos inferiores na hierarquia do judiciário[1].

Nesse cenário torna-se mais importante explorarmos o uso de precedentes. Para isso é importante explorar dados sobre seu uso para obtermos informações sobre como eles são utilizados nas decisões, assim como quais precedentes são muito citados e considerados como importantes pelos próprios decisores.

Para isso, este projeto oferece uma ferramenta de processamento de texto para extração de citações de precedentes no Superior Tribunal de Justiça (STJ) e no Supremo Tribunal Federal (STF). O programa pode ser encontrado no repositório público do GitHub: `https://github.com/joseluizn/extrator`. Baseamos nosso método no uso de expressões regulares para identificar padrões encontrados no formato de citação nas decisões dos tribunais.

A versão pública do projeto é uma adaptação da ferramenta utilizada em um repositório privado. Esta adaptação consistiu na remoção das funções que utilizavam conexão com um banco de dados que armazena informações das decisões, incluindo seu texto, adaptando as demais para lidar diretamente com arquivos salvados localmente.

Como parte do desenvolvimento expandimos o processo de extração para incluir também as classes e diferentes padrões de citação utilizados pelo STJ. Para isso, anotamos um conjunto de 80 decisões desse Tribunal selecionadas aleatoriamente entre as decisões publicadas entre 2008 e 2019 com o tema de Direito Societário. Essas decisões foram utilizados para medir a acurácia e o *recall* do programa, e os novos padrões identificados foram também incorporados ao programa. Além disso, a medição das métricas foi incorporada também como uma rotina de testes do programa.

Por fim, nossa ferramenta não possuía uma rotina de testes. Assim, desenvolvemos os testes apresentados aqui para verificar o funcionamento das funções e módulos do programa.

Antes de analisar melhor a ferramenta desenvolvida faremos uma breve explicação do conceito e uso do termo precedente.

Neste trabalho, o uso do termo precedente estará se referindo a decisões pretéritas publicadas por órgãos decisórios do judiciário que sejam citadas em decisões mais recentes. Cada um desses casos tem uma forma de identificação; para os tribunais superiores a forma mais comum é composta da classe e número dentro daquela classe. Por exemplo, uma decisão no caso Habeas Corpus 123 pode ser mencionada como HC 123 em uma nova decisão.

Contudo, esse uso pode não ser considerado conceitualmente preciso. Se-

[1] Artigo 927 da Lei 13.105/15

gundo Mitidiero (2018), precedentes são estruturados sobre os fatos jurídicos relevantes que compõe o caso em análise. Ainda, Marinori et al. (2018) afirmam que o fator decisivo para formação de precedente é a presença de "razões determinantes e suficientes claramente identificáveis".

Esta discussão visa apenas oferecer uma breve introdução ao conceito utilizado, e não encerra a discussão conceitual que existe ao redor do termo.

2 Trabalhos Relacionados

A quantidade de trabalhos jurídicos quantitativos é ainda bastante reduzida no Brasil, de forma que há raríssimas publicações que utilizem dados como os produzidos por nosso *software*. Essa característica se reflete também nos dados existentes sobre o sistema jurídico. Enquanto nos Estados Unidos bases como West Law[2] e Lexis Nexis[3] oferecem serviços e bases de dados jurídicas, que incluem dados como casos citados, não existem produtos semelhantes no Brasil.

Alguns trabalhos exploratórios já foram publicados com o objetido de produzir dados jurídicos. Estes utilizaram algoritmos mais sofisticados de aprendizado para extrair Entidades Nomeadas (Luz de Araujo et al., 2018). Contudo, a quantidade de dados anotados para treinamento e avaliação são muito reduzidos, sendo composto por apenas 66 decisões de diversos tribunais do país.

Trabalho semelhante foi realizado por Stemler (2019). Este, utilizou uma base de dados publicada pelo Conselho Nacional de Justiça contendo processos de diversos tribunais, explorando diversos modelos que pudessem sugerir bons resultados em uma pesquisa mais ampla.

Como comparação, utilizamos 80 decisões do STJ apenas para adaptar o extrator, que já havia sido aplicado ao STF, e verificar sua performance no tribunal

Além disso, Correia et al. (2019) utilizou os dados produzidos pelo programa apresentado aqui. Os autores realizaram uma análise exploratória a partir dos dados produzidos pelo algoritmo em decisões do Supremo Tribunal Federal

No âmbito internacional, existem algumas publicações com foco nesse tipo de informação. Fowler and Jeon (2008) e Leibon et al. (2018) são exemplos de trabalhos que partiram da extração de citações de precedentes e propuseram formas de ranquear os casos decididos pelo tribunal.

Além disso, Dozier et al. (2010) utilizaram um método semelhante em parte de sua proposta para Reconhecimento de Entidade Mencionada em diversos documentos jurídicos, incluindo decisões. Assim, nossa ferramenta está alinhada com o que já foi proposta pela literatura.

3 Extraindo Citações

Como mencionado acima, este *software* utiliza a estratégia de identificação de padrões com expressões regulares para extrair os dados textuais desejados. Isso é feito a partir da criação de *flags* que capturem as estruturas de citação conhecidas que utilizam a estrutura de classe processual e número designada pelo próprio Tribunal. A classe está associada ao tipo de ação, existindo uma lista quase exaustiva das opções possíveis, e a numeração é atribuída sequencialmente para cada classe.

[2] `https://legal.thomsonreuters.com/en/products/westlaw`

[3] `https://www.lexisnexis.com/en-us/home.page`

O algoritmo analisa todo o texto do arquivo recebido. O Supremo Tribunal Federal (assim como o STJ) possuí dois tipos de de decisão: (i) monocrática; e (ii) acórdãos. O primeiro são emitidas por ministros individualmente, nos casos previstos legalmente e pelo regimento interno do tribunal. A segunda é a decisão típica esperada quando pensamos na decisão de um tribunal, envolvendo a decisão de todos ou um subgrupo dos membros do tribunal. No geral, decisões monocráticas possuem uma extensão menor que acórdãos.

Algumas decisões monocráticas possuem apenas o texto contendo a decisão do caso, analisando as questões legalmente relevantes. Contudo, há também casos onde o texto inicia-se com o relatório, que contém uma breve descrição dos fatos e acontecimentos anteriores relevantes para a decisão. Essa é a estrutura básica da decisão do tribunal.

Já os acórdãos possuem uma estrutura mais elaborada. Seu documento começa com informações do processo, como ministro relator e , uma ementa, que sintetiza os argumentos jurídicos utilizados na decisão e enuncia a decisão final do tribunal (se foi de concessão, não concessão ou não conhecimento), seguido pelo relatório e a decisão. O tamanho médio dessas decisões, quando emitidas pelo Plenário do Tribunal é de pouco mais de 10 páginas (Hartmann et al., 2017)

Um exemplo da ementa da decisão pode ser encontrado na Figura 1.

O texto foi extraído de um Habeas Corpus, e estão destacadas 3 citações a outros Habeas Corpus. O formato das 3 citações é: CLASSE_PROCESSUAL NUM_PROCESSO/ORIGEM, onde origem é a sigla do estado de origem do processo.

Na Figura 2 constam exemplos simples de citações em uma decisão do STF

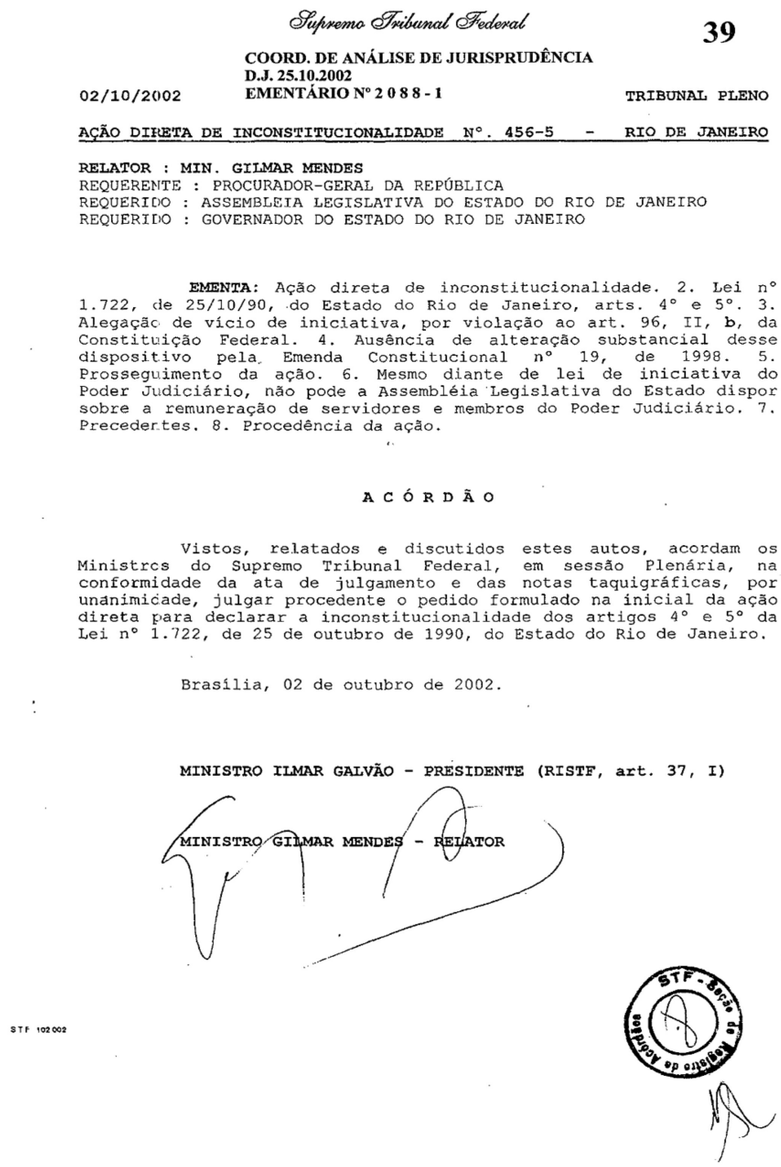
Outra consideração sobre estas citações é que todas estão no singular. O extrator possui um padrão para citações singulares, como o da Figura 2, e outro para citações no plural. Um exemplo de citações na estrutura pluralizada pode ser encontrado na Figura 3.

Esta citação em específico não seria capturada por nosso programa, já que apresenta uma informação extra, o ministro relator, entre cada um dos processos citados. Entretanto, podemos observar a estrutura desejada se ignorarmos essa informação. Ela pode ser estruturada da seguinte forma: PLURAL_CLASSE_PROCESSUAL NUM_PROCESSO(, NUM_PROCESSO) E NUM_PROCESSO. O elemento entre parênteses pode ser repetido múltiplas vezes.

Além desses elementos, é possível que alguns acessórios também estejam presentes. Por exemplo, um "N", "nº"ou "n."antes do algarismo que indica o número. Essa possibilidade também foi incluída na expressão regular criada.

Por fim, há ainda expressões para capturar menções específicas para citação de Súmulas, que não são bem suportadas pela estrutura apresentada. Isto porque, nesses casos, o número pode vir antes do uso da palavra súmula, como na expressão "Enunciado 691 de Súmula".

Outra informação que também é procurada pelo sistema é se há alguma referência ao tribunal responsável pelo processo citado. Isso é especialmente relevante uma vez que há uma gama de classes com o mesmo nome, tanto no STJ quanto no STF (*e.g.*, Habeas Corpus (HC) e Mandado de Segurança (MS)). Isto é feito de forma simples, buscando-se por uma menção ao nome do tribunal a uma distância considerada razoável da citação, se o nome de algum dos tribunais é encontrado assume-se que o processo é oriundo dele. Caso contrário, infere-se que seja do mesmo tribunal que emitiu a decisão.



Supremo Tribunal Federal 39

COORD. DE ANÁLISE DE JURISPRUDÊNCIA
D.J. 25.10.2002
02/10/2002 EMENTÁRIO Nº 2088-1 TRIBUNAL PLENO

AÇÃO DIRETA DE INCONSTITUCIONALIDADE Nº. 456-5 - RIO DE JANEIRO

RELATOR : MIN. GILMAR MENDES
REQUERENTE : PROCURADOR-GERAL DA REPÚBLICA
REQUERIDO : ASSEMBLEIA LEGISLATIVA DO ESTADO DO RIO DE JANEIRO
REQUERIDO : GOVERNADOR DO ESTADO DO RIO DE JANEIRO

EMENTA: Ação direta de inconstitucionalidade. 2. Lei nº 1.722, de 25/10/90, do Estado do Rio de Janeiro, arts. 4º e 5º. 3. Alegação de vício de iniciativa, por violação ao art. 96, II, b, da Constituição Federal. 4. Ausência de alteração substancial desse dispositivo pela Emenda Constitucional nº 19, de 1998. 5. Prosseguimento da ação. 6. Mesmo diante de lei de iniciativa do Poder Judiciário, não pode a Assembléia Legislativa do Estado dispor sobre a remuneração de servidores e membros do Poder Judiciário. 7. Precedentes. 8. Procedência da ação.

A C Ó R D Ã O

Vistos, relatados e discutidos estes autos, acordam os Ministros do Supremo Tribunal Federal, em sessão Plenária, na conformidade da ata de julgamento e das notas taquigráficas, por unanimidade, julgar procedente o pedido formulado na inicial da ação direta para declarar a inconstitucionalidade dos artigos 4º e 5º da Lei nº 1.722, de 25 de outubro de 1990, do Estado do Rio de Janeiro.

Brasília, 02 de outubro de 2002.

MINISTRO ILMAR GALVÃO - PRESIDENTE (RISTF, art. 37, I)

MINISTRO GILMAR MENDES - RELATOR

STF 102002

Figura 1: Exemplo de ementa de acórdão do STF na ADI 456.

4 Projeto Arquitetural

Esta seção apresenta alguns aspectos e informações da arquitetura do Extrator de Precedentes. O tópico começará com a visão de uso da ferramenta, seguida pela descrição dos componentes do programa. Finalmente, apresentaremos o processo de execução do programa.

4.1 Visão de Casos de Uso

A ferramenta apresentada aqui tem um uso bem delimitado. Seu objetivo é extrair citações de decisões do Superior Tribunal de Justiça ou do Supremo Tribunal Federal. Inicialmente, o programa recebe um arquivo .json contendo uma lista de arquivos que contenham as decisões que devem ser processadas.

O programa retorna um arquivo em formato *Comma Separated Values* (CSV). Este arquivo lista as citações encontradas em cada um dos arquivos analisados, contendo o ID da decisão original (o qual espera-se que seja o nome

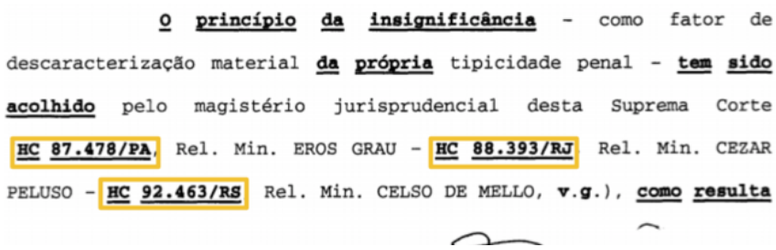

O princípio da insignificância - como fator de descaracterização material da própria tipicidade penal - tem sido acolhido pelo magistério jurisprudencial desta Suprema Corte HC 87.478/PA, Rel. Min. EROS GRAU - HC 88.393/RJ Rel. Min. CEZAR PELUSO - HC 92.463/RS Rel. Min. CELSO DE MELLO, v.g.), como resulta

Figura 2: Exemplos simples de citações de Habeas Corpus

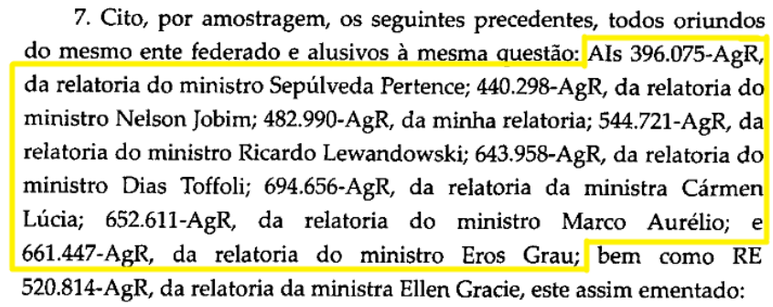

7. Cito, por amostragem, os seguintes precedentes, todos oriundos do mesmo ente federado e alusivos à mesma questão: AIs 396.075-AgR, da relatoria do ministro Sepúlveda Pertence; 440.298-AgR, da relatoria do ministro Nelson Jobim; 482.990-AgR, da minha relatoria; 544.721-AgR, da relatoria do ministro Ricardo Lewandowski; 643.958-AgR, da relatoria do ministro Dias Toffoli; 694.656-AgR, da relatoria da ministra Cármen Lúcia; 652.611-AgR, da relatoria do ministro Marco Aurélio; e 661.447-AgR, da relatoria do ministro Eros Grau; bem como RE 520.814-AgR, da relatoria da ministra Ellen Gracie, este assim ementado:

Figura 3: Exemplo de citações no plural

do arquivo) e as Classes, Números e Tribunais dos processos citados. Isso foi representado na Figura 4.

4.2 Visão de Componentes

Esta seção apresenta uma visão dos componentes que formam o extrator de precedentes. Sua estrutura como programa é simples, sendo formado por apenas 3 módulos de código, sendo um de funções auxiliares, além de um de configuração de *log*.

4.2.1 Diagrama do Programa

A Figura 5 apresenta um diagrama do código do Extrator de Precedentes. O diagrama UML não representa um diagrama de classe *per se*, uma vez que esse paradigma não é utilizado em nosso programa.

4.3 Visão de Processo

Esta seção apresenta uma visão do ffuncionamento do extrator de precedentes. Como o programa só apresenta uma funcionalidade os diagramas são simples, ilustrando o processo de extração de informação dos textos de decisão sob diferentes perspectivas.

4.3.1 Diagramas de Fluxo

Podemos representar o funcionamento utilizando um único diagrama de fluxo constante na Figura 6. A execução do programa se inicia com o *input* de um arquivo JSON contendo uma lista de *paths* para arquivos de texto que contenham decisões, como apresentado na documentação.

A partir das informações de cada arquivo, temos uma sequência relativamente simples. O arquivo é processado, identificando-se trechos textuais onde haja citações. Cada trecho é processado, quebrando-se em citações individuais para aqueles trechos onde a citação é feita na forma de plural. Então os trechos

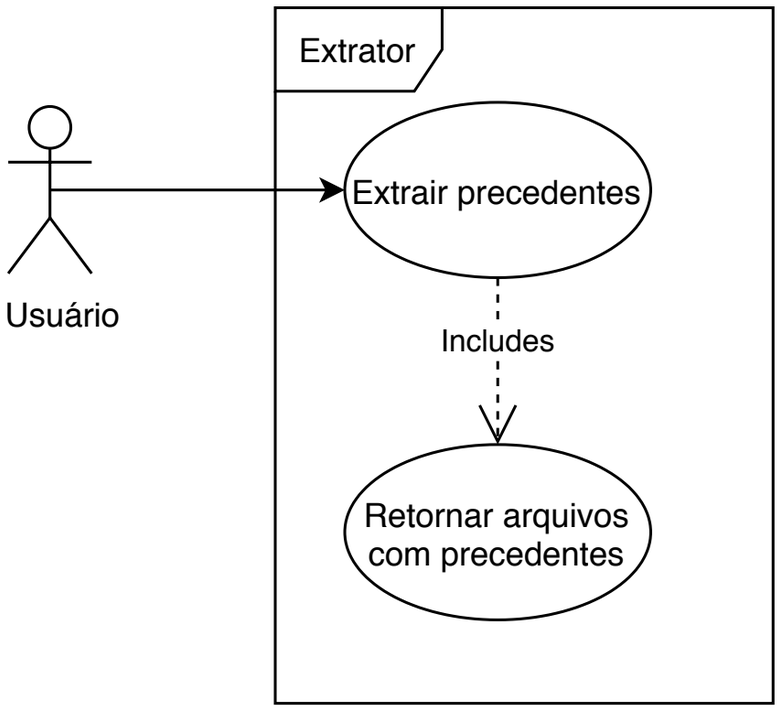


Figura 4: Diagrama de Casos de Uso

são normalizados para padronização e salvos no arquivo contendo o *output* do programa.

4.3.2 Diagramas de Sequência

A Figura 7 exibe o funcionamento do programa a partir de um diagrama de sequência. Além do usuário e do Extrator, é exibido como intermediário um gerenciador de multitarefas que gerencia a execução em paralelo da extração.

5 Testes

Os testes foram automatizados utilizando o módulo *Unittest*, nativo do Python[4]. Os testes podem ser encontrados nos arquivos .py, e estão armazenados na pasta `test` dentro da raiz do diretório. Os testes devem ser executados rodando o seguinte comando a partir de um terminal na raiz do diretório.

```
python -m unittest discover -s test
```

Os testes foram criados para testar individualmente cada função que compõe o programa, a partir de seu resultado esperado. Os arquivos de teste foram quebrados por módulo; assim, cada módulo possui seu módulo de testes, como mostra a Tabela 1.

Além disso, para testar a performance geral do programa, um dos arquivos de teste compara o resultado da extração do programa no estado atual com métricas anteriores, a partir de um conjunto pré-determinado de documentos do Superior Tribunal de Justiça. Este conjunto foi anotado manualmente, e a partir disso geramos uma medida de precisão e *recall* das extrações. Neste caso, o teste falha se apresentar um *score* inferior aos anteriores em alguma dessas medidas

[4] `https://docs.python.org/2/library/unittest.html`

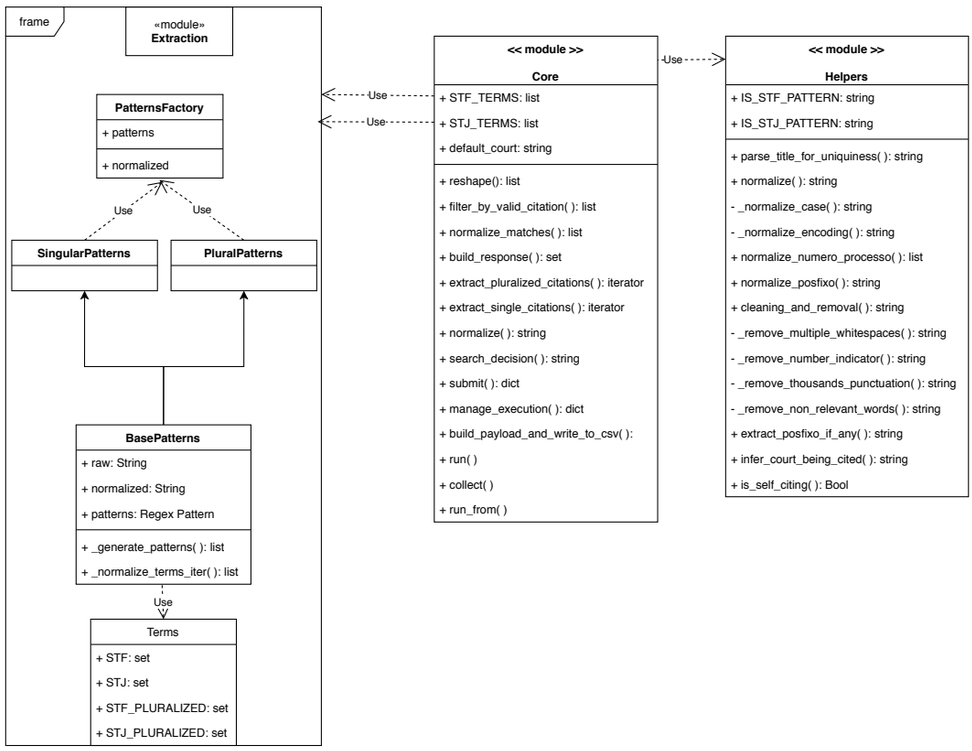


Figura 5: Diagrama do Extrator de Precedentes

Nome do Arquivo	**Descrição**
`tests_extraction`	1 – Testes do módulo *extraction.py* 2 – Testes focados na identificação e extração de trechos de citação de conteúdo textual.
`tests_core`	1 – Testes do módulo *core.py* 2 – Testes verificam se a sequência e funções que lidam com o *workflow* do programa estão retornando resultados esperados
`tests_helpers`	1 – Testes do módulo *helpers.py* 2 – Funções que fazem processamento individual de cada unidade de dados sendo processada
`tests_performance`	1 – Testa performance do extrator utilizando conjunto de decisões do STJ 2 – Analisa precisão e *recall* do algoritmo a partir de *data set* anotado manualmente de decisões e precedentes

Tabela 1: Rotinas de Teste do Extrator de Precedentes

6 Manual do Usuário

Esta seção descreve um passo-a-passo da utilização da ferramenta, desde sua instalação e pacotes necessários. Mostraremos o formato necessário para *input* dos dados no Extrator.

6.1 Instalação

O Extrator tem seu código feito em Python 2.7. O tutorial de instalação supõe que já há um ambiente desta linguagem instalado. Com isso, passamos aos pacotes necessários para execução do programa.

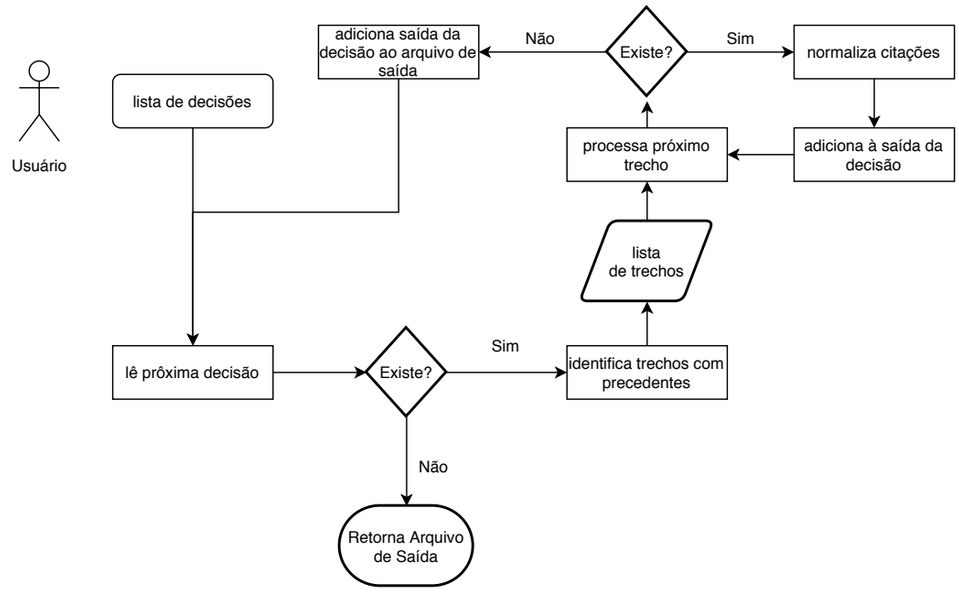


Figura 6: Diagrama de Fluxo do Extrator de Precedentes

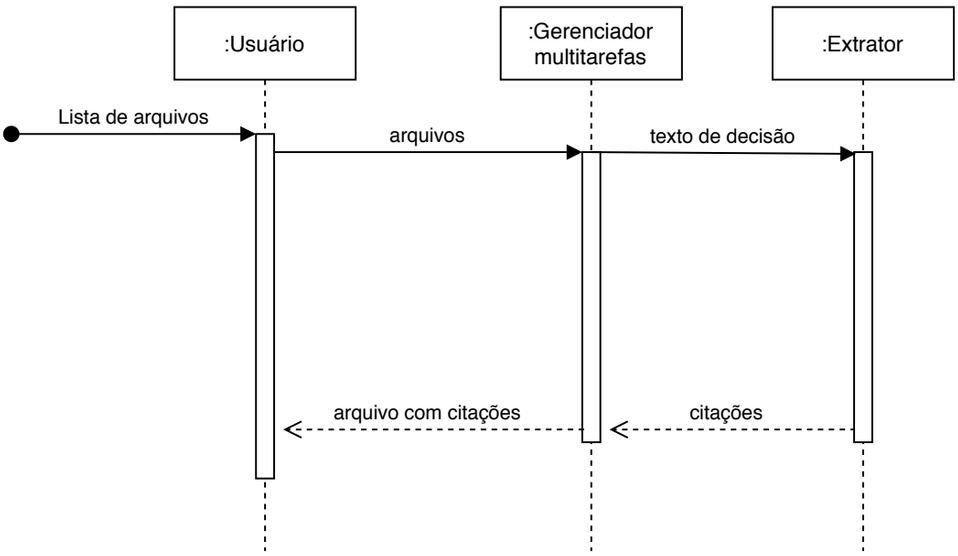


Figura 7: Diagrama de Sequência do Extrator de Precedentes

Para instalar os pacotes necessários basta instalar a lista de pacotes presente no arquivo `requirements.txt`. Para isso deve executar-se o comando `pip install -r requirements.txt` do repositório raiz do programa.

Isto instalará os seguintes pacotes:

- Numpy
- Pandas
- PathLib
- pkg-resources
- python-dateutil
- Pytz
- six

Alguns desses pacotes não são utilizados diretamente e são dependências ou do ambiente ou de outros pacotes utilizados.

6.2 Configuração

A configuração do programa é extremamente simples. Como parâmetros o programa espera apenas 3 argumentos, exibidos na Tabela 2

Nome do argumento	**Descrição**	**Obrigatório**
`--ids`	Arquivo de texto contendo listagem de *paths* em formato JSON de arquivos para serem processadas.	Sim
`--default_court` ou `-c`	Corte das quais arquivos estão sendo extraídos para inferir origem das citações.	Sim
`--n_cores`	Número de núcleos para processar arquivos paralelamente.	Não

Tabela 2: Argumentos que podem ser passados ao extrator

A seguir mostramos um exemplo de uma lista de decisões a serem extraídas, supondo que estas se encontram em uma pasta chamada `decisoes_stj` na raíz do projeto. A estrutura é a de um objeto lista em Python, iniciando-se e terminando com colchete. Além disso, cada arquivo deve ter seu nome envolto por aspas.

```
[
    "decisoes_stj/20081219_Ag_943418_7513484",
    "decisoes_stj/20081008_Ag_973204_7540554",
    "decisoes_stj/20080530_Ag_994603_7549473",
    "decisoes_stj/20080612_REsp_1061338_2632838",
    "decisoes_stj/20080805_Ag_965636_3916014",
    "decisoes_stj/20081212_REsp_1012999_2746074",
    "decisoes_stj/20080804_Ag_1057544_76275",
    "decisoes_stj/20081114_REsp_989746_1804800",
    "decisoes_stj/20080430_Ag_1015912_3959395",
    "decisoes_stj/20080804_REsp_1071589_2685477",
    "decisoes_stj/20091023_Ag_1205285_4053663",
    "decisoes_stj/20091016_REsp_1138831_2943894",
    "decisoes_stj/20100226_Ag_1184980_4140139",
    "decisoes_stj/20101203_Ag_1277119_4605305",
    "decisoes_stj/20170627_CC_152615_8331456",
    "decisoes_stj/20170802_CC_153267_8334312"
]
```

Suponha que este arquivo se chame `arquivo_teste.json`. Como as decisões listadas são do Superior Tribunal de Justiça, o comando para execução do programa seria:

```
python core.py --ids arquivo_teste.json -c STJ
```

Ainda, se quisermos estabelecer o número de núcleos (e tarefas) a serem executadas, podemos incluir no comando o argumento `--n_cores`. Suponha que queremos utilizar 3 núcleos, o comando seria:

```
python core.py --ids 1_id_mock.json -c STJ --n_cores 3
```

6.3 Saída do Programa

O *output* do programa é salvo no arquivo "out.csv". Um exemplo de saída está apresentado na Tabela 3. Nesse exemplo, o valor do Documento é o nome inicial do arquivo que o programa recebeu, e indica de onde a citação foi originada. As colunas seguintes identificam o processo citado. "Classe" corresponde a classe processual do caso citado; "Num" é o número designado pelo tribunal, que compõe o identificador único em conjunto com a classe; e Tribunal é o tribunal que emitiu a decisão citada (STF ou STJ).

Documento	Classe	Num	Tribunal
20170526_AREsp_1035422_8421733	resp	1234057	STJ
20170526_AREsp_1035422_8421733	resp	1507973	STJ
20080502_Ag_1016162_102373	sum.	5	STJ
20080502_Ag_1016162_102373	sum.	7	STJ
20080502_Ag_1016162_102373	resp	829835	STJ
20080502_Ag_1016162_102373	resp	208468	STJ
20080502_Ag_1016162_102373	eresp	237553	STJ
20080502_Ag_1016162_102373	resp	473159	STJ

Tabela 3: Exemplo de saída do Extrator

7 Considerações Finais e Trabalhos Futuros

Nesse documento apresentamos um programa criado para extrair precedentes de decisões dos dois principais tribunais de cúpula do país. O objetivo de extrair precedentes desses tribunais foi motivado pela sua importância dentro do sistema jurídico brasileiro, uma vez que são responsáveis por decidir em última instância a maior parte das questões jurídicas.

O programa foi inicialmente desenvolvido para o STF, tendo-se em mente que o STJ utilizava um padrão semelhante em suas citações. Como o método de extração utilizado é de identificação de padrões a partir de expressões regulares, o programa foi adaptado para funcionar também em decisões do Superior Tribunal de Justiça. A relevância do programa se encontra em produzir dados para uma área onde estes ainda são pouco explorados, tanto academicamente quanto para auxiliar na atuação de advogados e membros do judiciário.

Há três linhas a serem exploradas em trabalhos futuros. A primeira é adaptar e melhorar os padrões utilizados para aumentar a performance nos casos de uso já previstos, extraindo mais precedentes e evitando falsos positivos. Uma segunda possibilidade é expandir a mesma técnica para outros tribunais. Isto iria requerer um esforço de coleta e análise de dados para encontrar padrões textuais que sejam utilizados, que podem não ser os mesmos que utilizamos aqui.

Uma última possibilidade é aplicar outras estratégias para extrair as mesmas informações. Uma possibilidade, também utilizada na área de reconhecimento de Entidades Nomeadas, é o uso de modelos de aprendizado de máquina em dados anotados para a identificação dos mesmos padrões aqui codificados e exprimidos em regras manualmente. Um exemplo de uso dessa estratégia na mesma tarefa desempenhado por nosso programa pode ser encontrado em Leitner et al. (2019)

8 Anexos

8.1 Lista de Classes e Siglas Normalizadas

Essa seção do Anexo mostra a lista de classes utilizada na extração para cada tribunal, assim como sua sigla. As classes são mostradas em sua forma normalizada para *strings* em Python 2, formato ASCII.

8.1.1 Classes no Supremo Tribunal Federal

- “acao cautelar”: “ac”
- “acao civel originaria”: “aco”
- “acao declaratoria de constitucionalidade”: “adc”
- “acao direta de inconstitucionalidade por omissao”: “ado”
- “acao direta de inconstitucionalidade”: “adi”
- “acao originaria especial”: “aoe”
- “acao originaria”: “ao”
- “acao penal”: “ap”
- “acao rescisoria”: “ar”
- “adin”: “adi”
- “agravo de instrumento”: “ai”
- “apelacao civel”: “aci”
- “arguicao de descumprimento de preceito fundamental”: “adpf”
- “arguicao de impedimento”: “aimp”
- “arguicao de relevancia”: “arv”
- “arguicao de suspeicao”: “as”
- “carta rogatoria”: “cr”
- “comunicacao”: “cm”
- “conflito de atribuicoes”: “ca”
- “conflito de competencia”: “cc”
- “conflito de jurisdicao”: “cj”
- “enunciado sumular”: “sum.”
- “excecao da verdade”: “ev”
- “excecao de incompetencia”: “ei”
- “excecao de litispendencia”: “el”
- “excecao de suspeicao”: “es”
- “execucao penal”: “ep”
- “extradicao”: “ext”
- “habeas corpus”: “hc”
- “habeas data”: “hd”
- “inquerito”: “inq”
- “intervencao federal”: “if”
- “mandado de injuncao”: “mi”
- “mandado de seguranca”: “ms”
- “oposicao em acao civil originaria”: “oaco”
- “peticao avulsa”: “peta”
- “peticao”: “pet”
- “prisao preventiva para extradicao”: “ppe”
- “processo administrativo”: “pa”
- “proposta de sumula vinculante”: “psv”
- “queixa-crime”: “qc”
- “reclamacao”: “rcl”
- “recurso crime”: “rc”
- “recurso extraordinario com agravo”: “are”
- “recurso extraordinario”: “re”

- “recurso ord. em mandado de seguranca”: “rms”
- “recurso ordinario em habeas corpus”: “rhc”
- “recurso ordinario em habeas data”: “rhd”
- “recurso ordinario em mandado de injuncao”: “rmi”
- “representacao”: “rp”
- “revisao criminal”: “rvc”
- “sentenca estrangeira contestada”: “sec”
- “sentenca estrangeira”: “se”
- “sumula vinculante”: “sum. vinc.”
- “sumula”: “sum.”
- “suspensao de liminar”: “sl”
- “suspensao de seguranca”: “ss”
- “suspensao de tutela antecipada”: “sta”

8.2 Classes no Superior Tribunal de Justiça

- “acao de improbidade administrativa”: “aia”
- “acao penal”: “apn”
- “acao rescisoria”: “ar”
- “agravo”: “ag”
- “agravo de instrumento em rhc para stf”: “ag/rhc”
- “agravo de instrumento em rms para stf”: “ag/rms”
- “agravo de instrumento para o presidente”: “agpres”
- “agravo de instrumento para stf”: “ag/re”
- “agravo de instrumento”: “ag”
- “agravo em recurso especial”: “aresp”
- “apelacao civel”: “ac”
- “carta rogatoria”: “cr”
- “comunicacao”: “com”
- “conflito de atribuicao”: “cat”
- “conflito de competencia”: “cc”
- “embargos de divergencia em agravo em recurso especial”: “earesp”
- “embargos de divergencia em agravo”: “eag”
- “embargos de divergencia em resp”: “eresp”
- “embargos de divergencia em rms”: “erms”
- “enunciado sumular”: “sum.”
- “excecao da verdade”: “exverd”
- “excecao de impedimento”: “eximp”
- “excecao de suspeicao”: “exsusp”
- “execucao em acao rescisoria”: “exear”
- “execucao em mandado de seguranca”: “exems”
- “habeas corpus”: “hc”
- “habeas data”: “hd”
- “homologacao de decisao estrangeira”: “hde”
- “incidente de deslocamento de competencia”: “idc”
- “inquerito”: “inq”
- “interpelacao judicial”: “ij”
- “intervencao federal”: “if”
- “mandado de injuncao”: “mi”
- “mandado de seguranca”: “ms”
- “medida cautelar”: “mc”
- “medidas investigativas sobre organizacoes criminosas”: “misoc”
- “noticia-crime”: “nc”
- “pedido de tutela provisoria”: “tp”
- “pedido de uniformizacao de interpretacao de lei”: “puil”

- “peticao”: “pet”
- “precatorio”: “prc”
- “reclamacao”: “rcl”
- “recurso em habeas corpus”: “rhc”
- “recurso em mandado de seguranca”: “rms”
- “recurso especial”: “resp”
- “recurso ordinario”: “ro”
- “representacao”: “rp”
- “requisicao de pequeno valor”: “rpv”
- “revisao criminal”: “rvcr”
- “sentenca estrangeira contestada”: “sec”
- “sentenca estrangeira”: “se”
- “sindicancia”: “sd”
- “sumula”: “sum.”
- “suspensao de liminar e de sentenca”: “sls”
- “suspensao de liminar”: “sl”
- “suspensao de seguranca”: “ss”
- “suspensao de tutela antecipada”: “sta”

8.3 Testes

Um exemplo de cabeçalho de arquivo de teste pode ser encontrado nesta sessão, retirado do módulo `tests_core.py`.

```
#!/usr/bin/env python
# coding=latin-1

import re
import unittest

from extraction import PatternsFactory
import helpers
import core


class CoreTests(unittest.TestCase):
    """
    Tests function from core module.
    """
```

Após executar os testes, o *output* retornado é que não houve falha na execução de 35 testes, como pode ser visto na Imagem 8.

```
.....................................
----------------------------------------------------------------------
Ran 35 tests in 2.715s

OK
```

Figura 8: Saída de execução de testes

Referências

Fernando A. Correia, José Luiz Nunes, Guilherme F. C. F. de Almeida, Alexandre A. A. Almeida, and Hélio Lopes. An exploratory analysis of precedent relevance in the brazilian supreme court rulings. In *Proceedings of the ACM Symposium on Document Engineering 2019*, DocEng '19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450368872. doi: 10.1145/3342558.3345416. URL https://doi.org/10.1145/3342558.3345416.

Christopher Dozier, Ravikumar Kondadadi, Marc Light, Arun Vachher, Sriharsha Veeramachaneni, and Ramdev Wudali. Named entity recognition and resolution in legal text. In Enrico Francesconi, Simonetta Montemagni, Wim Peters, and Daniela Tiscornia, editors, *Semantic Processing of Legal Texts*, volume 6036 of *Lecture Notes in Computer Science*, page 27–43. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-12836-3. doi: 10.1007/978-3-642-12837-0_2. URL http://link.springer.com/10.1007/978-3-642-12837-0_2.

James H. Fowler and Sangick Jeon. The authority of supreme court precedent. *Social Networks*, 30(1):16–30, Jan 2008. ISSN 0378-8733.

Ivar Alberto Hartmann, Guilherme da Franca C. Fernandes de Almeida, Beatriz Nunes Valim, Clarissa Emanuel Leão Lima, Gabriel Borges Mariano, Larissa de Lima e Campos, and José Luiz Nunes. A influência da tv justiça no processo decisório do stf. *Revista de Estudos Empíricos em Direito*, 4(3), nov. 2017. doi: 10.19092/reed.v4i3.186. URL https://revistareed.emnuvens.com.br/reed/article/view/186.

Greg Leibon, Michael Livermore, Reed Harder, Allen Riddell, and Dan Rockmore. Bending the law: geometric tools for quantifying influence in the multinetwork of legal opinions. *Artificial Intelligence and Law*, 26(2):145–167, Jun 2018. ISSN 1572-8382.

Elena Leitner, Georg Rehm, and Julian Moreno-Schneider. Fine-grained named entity recognition in legal documents. In Maribel Acosta, Philippe Cudré-Mauroux, Maria Maleshkova, Tassilo Pellegrini, Harald Sack, and York Sure-Vetter, editors, *Semantic Systems. The Power of AI and Knowledge Graphs*, Lecture Notes in Computer Science, page 272–287. Springer International Publishing, 2019. ISBN 978-3-030-33220-4. doi: 10.1007/978-3-030-33220-4_20.

Pedro Henrique Luz de Araujo, Teófilo E. de Campos, Renato R. R. de Oliveira, Matheus Stauffer, Samuel Couto, and Paulo Bermejo. Lener-br: A dataset for named entity recognition in brazilian legal text. In Aline Villavicencio, Viviane Moreira, Alberto Abad, Helena Caseli, Pablo Gamallo, Carlos Ramisch, Hugo Gonçalo Oliveira, and Gustavo Henrique Paetzold, editors, *Computational Processing of the Portuguese Language*, volume 11122 of *Lecture Notes in Computer Science*, page 313–323. Springer International Publishing, 2018. ISBN 978-3-319-99721-6. doi: 10.1007/978-3-319-99722-3_32. URL http://link.springer.com/10.1007/978-3-319-99722-3_32.

Luiz Guilherme Marinori, Sérgio Cruz Arenhart, and Daniel Mitidiero. *Código de Processo Civil Comentado*. Revista dos Tribunais, SP, 4 edition, 2018.

Daniel Mitidiero. *Cortes Superiores e Cortes Supremas: do controle à interpretação, da jurisprudência ao precedente*. Revista dos Tribunais, SP, 4 edition, 2018.

Igor Tadeu Silva Viana Stemler. Identificação de precedentes judiciais por agrupamento utilizando processamento de linguagem natural. Master's thesis, Universidade de Brasília, Jul 2019. URL https://repositorio.unb.br/handle/10482/36866. Accepted: 2020-02-10T18:37:19Z.