DOCUMENTOS

185

ISSN 2764-2488
Dezembro/2022

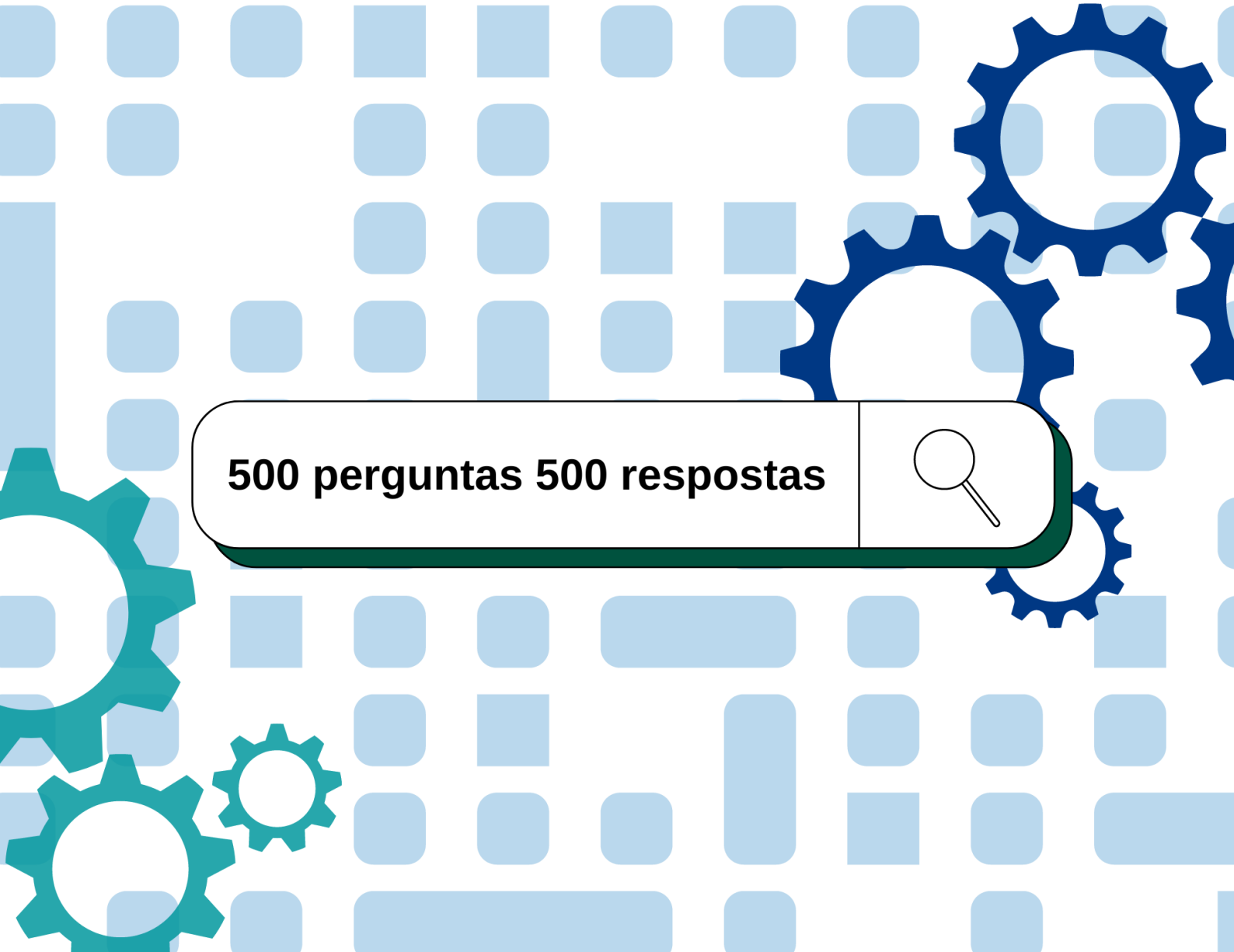
OBJETIVOS DE DESENVOLVIMENTO SUSTENTÁVEL

17 PARCERIAS E MEIOS DE IMPLEMENTAÇÃO

OBJETIVOS DE DESENVOLVIMENTO SUSTENTÁVEL

12 CONSUMO E PRODUÇÃO RESPONSÁVEIS

Consulta por Documento e Incorporação de Glossários em um Mecanismo de Busca para a Coleção 500 Perguntas 500 Respostas



ISSN 2764-2488
Dezembro/2022

Empresa Brasileira de Pesquisa Agropecuária
Embrapa Agricultura Digital
Ministério da Agricultura, Pecuária e Abastecimento

DOCUMENTOS 185

Consulta por Documento e Incorporação de Glossários em um Mecanismo de Busca para a Coleção 500 Perguntas 500 Respostas

Glauber José Vaz

Embrapa Agricultura Digital
Campinas, SP
2022

Embrapa Agricultura Digital
Av. Dr. André Tosello, nº 209 - Campus da Unicamp,
Barão Geraldo - Campinas, SP
CEP. 13083-886 - Fone: +55 (19) 3211-5700

www.embrapa.br/agricultura-digital
www.embrapa.br/fale-conosco/sac

Revisão de texto
Adriana Farah Gonzalez

Normalização bibliográfica
Carla Cristiane Osawa

Projeto gráfico da coleção
Carlos Eduardo Felice Barbeiro

Editoração eletrônica
Letícia Mathias do Amaral Campos

Foto da capa
Letícia Mathias do Amaral Campos

1ª edição
Publicação digital - PDF (2022)

Dados Internacionais de Catalogação na Publicação (CIP)
Embrapa Informática Agropecuária

Vaz, Glauber José.
Consulta por documento e incorporação de glossários em um mecanismo de busca para a Coleção 500 perguntas 500 respostas / Glauber José Vaz. - Campinas : Embrapa Agricultura Digital, 2022.
PDF (49 p.) : il. color. - (Documentos / Embrapa Agricultura Digital, ISSN 2764-2488 ; 185).

1. Tecnologia da informação. 2. Recuperação da informação. 3. API Responde Agro. I. Título. II. Embrapa Agricultura Digital. III. Série.

CDD (21. ed.) 004

Carla Cristiane Osawa (CRB-8/10421)

Autor

Glauber José Vaz
Bacharel e mestre em Ciência da Computação, analista da Embrapa Agricultura Digital, Campinas, SP

Apresentação

Mecanismos de busca fazem parte do dia a dia de qualquer usuário da internet. Além de ferramentas de uso geral, como aquelas que envolvem buscas no conteúdo disponível na web, mecanismos específicos também são importantes para auxiliar os públicos mais restritos em consultas a conteúdo mais direcionado.

A Empresa Brasileira de Pesquisa Agropecuária (Embrapa) cria muito conhecimento no domínio da Agropecuária Tropical. Um bom exemplo é a coleção "500 Perguntas 500 Respostas", conjunto de livros que reúnem perguntas selecionadas e respondidas pelo corpo técnico da Empresa em diferentes temas. Para se obter maior alcance desse conhecimento, é fundamental que ferramentas digitais auxiliem o público a encontrar as informações de seu interesse de maneira simples e contextualizada.

O projeto "EmbrapaIZagroSearch" teve como objetivo geral disponibilizar um mecanismo de busca, acessível via API, que recupere informações das obras "500 Perguntas 500 Respostas". Um dos resultados previstos no projeto visa oferecer melhor experiência ao usuário na realização de consultas, por meio da criação de métodos envolvendo algoritmos de inteligência artificial que exploram novas formas de apresentação do conteúdo, relações entre os conteúdos das obras e relações entre as obras e dados complementares, como glossários.

Neste trabalho, há o relato do resultado alcançado pela execução da atividade "Exploração de algoritmos de inteligência artificial para se obter melhores resultados nas buscas realizadas pelos usuários". O resultado é a metodologia científica descrita no projeto por "novos métodos envolvendo algoritmos

de inteligência artificial aplicados ao conteúdo das obras ‘500 Perguntas, 500 Respostas’, de maneira a explorar: a) novas formas de apresentação do conteúdo (ex: diferentes elementos visuais); b) relações entre os conteúdos das obras (ex: grupos de perguntas e respostas com elementos em comum); e c) relações entre as obras e dados complementares (ex: glossário), para oferecer melhor experiência ao usuário ao realizar consultas no mecanismo de busca”.

Stanley Robson Medeiros de Oliveira
Chefe-geral da Embrapa Agricultura Digital

Sumário

Introdução

A API Responde Agro é resultado do projeto "EmbrapaIZagroSearch: um mecanismo de busca para o conteúdo da coleção 500 Perguntas 500 Respostas". Ligado ao portfólio de "Automação e Agricultura de Precisão e Digital" da Empresa, esse projeto contribui para o desafio de inovação relacionado a viabilizar soluções digitais de suporte à análise de dados e à tomada de decisão. Seu principal objetivo foi construir um mecanismo de busca para o conteúdo da coleção e disponibilizá-lo via API na plataforma AgroAPI[1] da Embrapa, que oferece dados, informações e modelos agropecuários gerados pela Empresa por meio de APIs.

A Embrapa já vem disponibilizando APIs por meio da AgroAPI tanto para uso interno, como para negócios com organizações parceiras ou acesso aberto, pois facilita o estabelecimento de acordos entre organizações e viabiliza maior alcance dos resultados obtidos pela Empresa e seus parceiros (Vaz et al., 2017a). Neste caso, o ativo foi codesenvolvido pela Embrapa e uma startup de agricultura digital, a IZagro. O uso da AgroAPI auxiliou no estabelecimento desta parceria.

A API Responde Agro possibilita a oferta do conteúdo da coleção "500 Perguntas 500 Respostas" (Embrapa, 2022b) de maneira que outras instituições possam embutí-lo em suas próprias soluções digitais. Basicamente, a API retorna um conjunto ordenado de pares pergunta-resposta extraídos das obras da coleção que foram indexadas e que estão relacionados a uma consulta realizada pelo usuário por meio de um texto fornecido como entrada. As consultas podem estar no escopo de apenas um livro da coleção ou no de todas as obras indexadas. Ainda é possível explorar recursos de autocompletar, em que o campo de busca é preenchido automaticamente enquanto o usuário digita sua consulta.

O sistema de recuperação envolvido na API Responde Agro organiza cada par de pergunta e resposta como um documento. A Figura 1 mostra o exemplo dos campos do documento referente à pergunta 221 do livro sobre feijão-caupi. Os campos identificam o número da pergunta, os textos da pergunta

[1] Disponível em: https://www.agroapi.cnptia.embrapa.br/portal/.

e da resposta, o capítulo do livro correspondente, o título do livro e um identificador para ele, seu ano de publicação e os links para as versões digitais do livro em formatos Epub e PDF.


```
{"question_number": 221,
"question":"Qual é a importância do
controle preventivo de plantas da-
ninhas na cultura do feijão-caupi?",
"answer":"<p>O controle preventivo
tem a finalidade de impedir a entrada
e a disseminação de espécies daninhas
nas áreas cultivadas com o feijão-
-caupi onde elas decididamente ainda
não estão presentes.</p>", "chapter":
"Plantas Daninhas", "book": "Feijão-
-caupi", "book_id": "feijao-caupi",
"year": 2017, "epub": "http://ainfo.
cnptia.embrapa.br/digital/bitstream/
item/166086/1/-files-500p500r-feijao-
-caupi.epub", "pdf": "http://ainfo.
cnptia.embrapa.br/digital/bitstream/
item/166168/1/500P500R-Feijao-caupi.
pdf"}
```


Figura 1. Exemplo de documento indexado no mecanismo de busca

Todo mecanismo de busca pode ser melhorado por meio da evolução dos algoritmos e da adição de novos recursos, inclusive explorando métodos de inteligência artificial (IA) para melhorar a experiência dos usuários. Neste trabalho, algoritmos foram adaptados para testar novas funcionalidades que podem ser agregadas à API Responde Agro. O objetivo foi de introduzir algumas possibilidades de melhorias, já que o projeto todo foi desenvolvido em apenas 18 meses e esta etapa foi a última a ser executada entre as relacionadas à tecnologia desenvolvida. A ênfase foi dada a métodos simples que podem gerar valor ao ativo. Mais especificamente, foram consideradas, conforme previsto no projeto EmbrapaIZagroSearch, abordagens envolvendo novas formas de apresentação do conteúdo, relações entre os conteúdos das obras e relações entre as obras e dados complementares, como glossários. Esses recursos foram implementados explorando-se funcionalidades já oferecidas pela tecnologia Elasticsearch (Elastic, 2022), base da API Responde Agro.

Uso de Glossários

É muito comum, durante o uso de um sistema de recuperação de informação, haver dúvidas em relação à definição correta de termos presentes em documentos que constam nos resultados de busca. O acoplamento de um glossário a uma ferramenta de busca possibilita uma experiência mais rica ao usuário ao exibir as definições de termos com os quais ele se depara em um contexto em que já está em busca de informações confiáveis.

Vaz et al. (2017b) construíram um recurso de visualização de glossário em um sistema de recuperação de informação, a fim de exibir as definições de todos os termos presentes nos documentos selecionados e que constam no glossário utilizado no sistema. Os autores ainda fornecem alguns detalhes da implementação, mas Vaz (2017) também utiliza essa abordagem e apresenta em maiores detalhes a implementação deste recurso. O presente trabalho se baseia nessa implementação para acoplar um glossário ao sistema construído no escopo deste projeto. A Figura 2 mostra a configuração do mapeamento para o índice do glossário. Ela tem diferenças em relação aos trabalhos citados devido a diferenças nas aplicações consideradas, em detalhes de implementação e nas versões utilizadas da tecnologia Elasticsearch.

```
PUT glossary
{
  "settings": {
    "analysis": {
      "filter": {
        "one_whitespace":{
          "type": "pattern_replace",
          "pattern": "\\s{2,}",
          "replacement": " "
        },
        "remove_hyphen":{
          "type": "pattern_replace",
          "pattern": "-",
          "replacement": " "
        }
      },
      "tokenizer": {
        "semicolon_keyword":{
          "type":"pattern",
          "pattern":"[*;]+"
        }
      },
      "analyzer": {
        "glossary_analyzer": {
          "tokenizer": "semicolon_keyword",
          "filter": ["asciifolding", "remove_hyphen", "lowercase",
                     "trim", "one_whitespace"]
        }
      }
    }
  },
  "mappings": {
    "properties": {
      "term":{
        "type": "text",
        "analyzer": "glossary_analyzer"
      },
      "definition":{
        "type": "text",
        "index": false
      },
      "glossary":{
        "type":"keyword"
      }
    }
  }
}
```

Figura 2. Mapeamento para o índice do glossário

Neste mapeamento, cada documento possui três campos: o termo (“term”), sua definição (“definition”) e o identificador do glossário a que pertence o termo (“glossary”). Este último é do tipo “keyword” pois o identificador é usado conforme fornecido, sem processamento algum. O campo “definition” é do tipo texto e também não é processado porque não é indexado. Por isso, sua propriedade “index” tem valor “false”. Isso significa que consultas não são feitas em função deste campo. Ele apenas fornece informação de resultado. Finalmente, o campo “term” é processado pelo analisador “glossary_analyzer”, tanto na indexação quanto na busca. A Figura 3 mostra a indexação do termo “Irrigação gota a gota” e seu sinônimo “Irrigação por gotejamento” associado à sua definição e ao glossário cujo identificador é “recursos_hidricos”.

```
POST glossary/_doc/recursos_hidricos_002
{
  "term":"Irrigação gota a gota; Irrigação por gotejamento",
  "definition":"Sistema de irrigação em que a água é aplicada
ao solo  de maneira lenta, constante e de baixa-pressão
através de gotejadores, formando um padrão de umedecimento
chamado bulbo úmido.",
  "glossary":"recursos_hidricos"
}
```

Figura 3. Exemplo de indexação de um termo que apresenta sinônimo

A Figura 2 mostra a definição do analisador “glossary_analyzer”, composto pelo tokenizer “semicolon_keyword”, também definido no mapeamento, e pelos filtros “asciifolding”, “remove_hyphen”, “lowercase”, “trim” e “one_whitespace”. O tokenizer usado separa tokens usando o caractere ‘;’ como separador. Portanto, o texto “Irrigação gota a gota; Irrigação por gotejamento”, por exemplo, gera a lista de tokens [“Irrigação gota a gota”, “Irrigação por gotejamento”]. Isso possibilita indexar termos sinônimos com uma mesma definição. Cada token é considerado na sua totalidade como um texto do tipo “keyword”, de maneira que “Irrigação por gotejamento” constitui um token, mas “Irrigação” ou “gotejamento” não constituem tokens. A consulta da Figura 4 permite testar a indexação. Com o termo “Irrigação por gotejamento”, a busca encontra o documento indexado na Figura 3, assim como para o texto “Irrigação gota a gota”. No entanto, se o texto fosse “Irrigação”, “gotejamento”, “gota” ou “Irrigação gota a gota por gotejamento”, nenhum documento seria encontrado. Se em vez de “term”, o campo “definition” fosse usado na consulta da Figura 4, um erro ocorreria porque este campo não é indexado, e portanto, não pode ser alvo de busca.

```
GET glossary/_search
{
  "query": {
    "match": {
      "term": "Irrigação por gotejamento"
    }
  }
}
```

Figura 4. Consulta à definição do termo “Irrigação por gotejamento” no índice de glossários

O “glossary_analyzer” ainda apresenta quatro filtros de tokens, que modificam os tokens gerados para melhorar o mecanismo de busca. O primeiro, “asciifolding”, elimina diacríticos, de maneira que uma busca por “Irrigacao por gotejamento”, por exemplo, tenha sucesso, mesmo substituindo-se ‘ç’ por ‘c’ e ‘ã’ por ‘a’. O “remove_hyphen” elimina hifens, o que é importante nesta aplicação porque o analisador padrão que processa perguntas e respostas elimina hifens, de maneira que ocorrências de “lavoura-pecuária”, por exemplo, geram tokens “lavoura” e “pecuária” em sequência, em vez de “lavoura-pecuária”. Assim, para identificar os termos do vocabulário utilizado nos textos, os hifens precisam ser eliminados, considerando, por exemplo, “lavoura pecuária”, neste caso citado, sem uso do hifen. Já o filtro “lowercase” possibilita o sucesso de buscas sem diferenciar minúsculas de maiúsculas, como em “irrigação” em vez de “Irrigação”. O filtro “trim” desconsidera os espaços em branco nas extremidades de um termo como em “ irrigacao ”, e o “one_whistespace” substitui sequências de espaço em branco por um único caractere, de maneira que “irrigacao por gotejamento”, com quatro espaços em branco, por exemplo, também seja encontrado em uma busca no glossário.

Uma vez que todos os termos dos glossários utilizados sejam indexados, faz-se necessário um mecanismo que possibilite identificar nos documentos da aplicação os termos que estão presentes nesse índice de glossário. Portanto, o mapeamento dos documentos também precisa ser modificado. No caso da aplicação envolvida com a API Responde Agro, os termos serão identificados apenas nos campos correspondentes aos textos das perguntas e das respostas. Então esses dois campos precisam ser alterados para incluir este recurso de glossário. A Figura 5 mostra as alterações necessárias para sua implementação em um sistema que já trata de perguntas e respostas.

```
PUT responde500
{
  “settings”: {
    “index” : {
      “max_shingle_diff” : 10
    },
    “analysis”: {
      “filter”: {
        :
        “shingle_term”: {
          “type”: “shingle”,
          “min_shingle_size”: 2,
          “max_shingle_size”: 10,
          “output_unigrams”: true
        },
        “terms_on_file”: {
          “type”:”keep”,
          “keep_words_path”:”vocabulary.txt”
        }
      },
      “analyzer”: {
        :
        “glossary_doc_analyzer”:{
          “tokenizer”: “uax_url_email”,
          “filter”: [“asciifolding”, “lowercase”, “trim”,
                     “shingle_term”, “terms_on_file”]
        }
      }
    }
  },
  “mappings”: {
    “properties”: {
      “question”:{
        “copy_to”: “glossary”,
        :
      },
      “answer”:{
        “copy_to”: “glossary”,
        :
      },
      “glossary”:{
        “type”: “text”,
        “term_vector”: “yes”,
        “analyzer”: “glossary_doc_analyzer”
      }
    }
  }
}
```

Figura 5. Mapeamento para documentos considerando a identificação de termos de glossários.

As propriedades referentes aos textos das perguntas e das respostas, respectivamente “question” e “answer” têm seu conteúdo copiado para outra propriedade chamada “glossary” por meio do parâmetro “copy_to” do mapeamento. Assim, “glossary” é preenchido com valores obtidos dos campos “question” e “answer”, que são analisados pelo “glossary_doc_analyzer”. Este analisador é utilizado tanto na indexação quanto na busca, e permite a recuperação dos termos que fazem parte de um documento cujas definições são fornecidas pelos glossários considerados. O parâmetro “term_vector” com valor “yes” possibilita que os termos do campo “glossary” sejam armazenados e recuperados posteriormente.

A ideia deste analisador é de gerar uma lista que contenha apenas os termos que estão definidos no glossário utilizado. Supondo que este analisador seja usado no texto “Os principais elementos climáticos que influenciam a produtividade de grãos do feijão-caupi são precipitação pluviométrica, temperatura do ar e radiação solar” e que o glossário utilizado inclua apenas os seguintes termos desse texto: “produtividade”, “precipitação pluviométrica” e “radiação solar”, então, para este texto de entrada, o analisador “glossary_doc_analyzer” geraria como resultado a lista de termos [“produtividade”, “precipitação pluviométrica”, “radiação solar”], ignorando os demais elementos do texto de entrada. Ele funciona, portanto, como um identificador dos termos de glossário no conteúdo.

O analisador utiliza um *tokenizer* padrão “uax_url_email” para separar a cadeia de caracteres em tokens. Por exemplo, o processamento do texto “Cultivares, melhoramento genético e biotecnologia” gera a lista de tokens [“Cultivares”, “melhoramento”, “genético”, “e”, “biotecnologia”]. Os filtros de tokens utilizados realizam as seguintes transformações:

a) “asciifolding”: remove diacríticos. Exemplo: “genético” é substituído por “genetico”;

b) “lowercase”: transforma todas as letras em minúsculas. Exemplo: “Cultivares” é substituído por “cultivares”;

c) “trim”: elimina espaços em branco nas extremidades dos textos. Exemplo: “ texto ” seria substituído por “texto”;

d) “shingle_term”: gera termos compostos por tokens em sequência. Neste caso, continua gerando os tokens individualmente, por conta do valor “true” para o parâmetro “output_unigrams”, e cria termos compostos envolvendo de dois a dez tokens, definidos por “min_shingle_size” e “max_shingle_size”. A configuração para “max_shingle_diff” se faz necessária porque, por padrão, a tecnologia não aceita valores para “min_shingle_size” e “max_shingle_size” com diferença tão ampla. Exemplo: [“cultivares”, “melhoramento”, “genetico”, “e”, “biotecnologia”] geraria [“cultivares”, “cultivares melhoramento”, “cultivares melhoramento genetico”, “cultivares melhoramento genetico e”, “cultivares melhoramento genetico e biotecnologia”, “melhoramento”, “melhoramento genetico”, “melhoramento genetico e”, “melhoramento genetico e biotecnologia”, “genetico”, “genetico e”, “genetico e biotecnologia”, “e”, “e biotecnologia”, “biotecnologia”];

e) “terms_on_file”: mantém na lista de tokens apenas os termos presentes no glossário determinado por “vocabulary.txt”. Supondo que o conteúdo deste arquivo seja o exibido na Figura 6 e sua entrada seja o resultado apresentado no exemplo do filtro “shingle_term”, a saída do “terms_on_file” seria [“melhoramento genetico”, “biotecnologia”], com apenas os dois termos que aparecem no glossário.

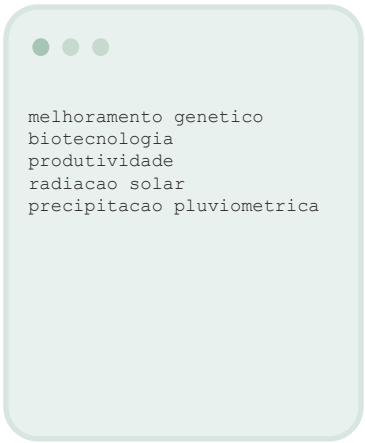


Figura 6. Lista de termos presentes no glossário.

O “glossary_doc_analyzer” do índice “responde500” não inclui o filtro “remove_hyphen” como o “glossary_analyzer” utilizado no índice “glossary” porque seu *tokenizer* padrão utiliza o hifen como um separador de tokens, e, assim, já remove hifens dos textos.

O analisador “glossary_doc_analyzer”, portanto, recebe um texto de entrada e retorna a lista de termos desse texto que está presente no glossário utilizado.

Os índices “glossary” e “responde500”, cujos mapeamentos são exibidos, respectivamente, nas Figuras 2 e 5 são independentes entre si. No entanto, é fundamental que tenham elementos em comum para o funcionamento adequado do recurso de exibição de glossário no mecanismo de busca. Os termos indexados em “glossary” por meio de requisições POST à API, conforme Figura 3, precisam ser os mesmos que constam no arquivo “vocabulary.txt” utilizado para localizar os termos do glossário no índice “responde500”.

Para garantir a consistência dos dados, um script em Python foi desenvolvido para a geração deste arquivo em função do índice já estabelecido em “glossary”. A Figura 7 mostra esse script, que executa passos similares aos utilizados no processamento do analisador “glossary_analyzer”: “asciifolding”, “lowercase”, “trim” e “one_whitespace”. O primeiro inclui em sua função a remoção de hifen da mesma maneira que o filtro “remove_hyphen”.

```
# Este script gera a lista de termos indexados em um índice de
# Elasticsearch referente a um glossário e disponível em *url*.
# Essa lista é escrita no arquivo *filename*.

import requests
import json
import re

url = "http://localhost:9200/glossary/_search"
filename = "vocabulary.txt"

# https://www.codegrepper.com/code-examples/python/How+do+I+remove+
# diacritics+%28accents%29+from+a+string+python
# It also removes hyphen
def asciifolding(text):
    import unicodedata
    try:
        text = unicode(text, 'utf-8')
    except NameError:
        pass
    text = unicodedata.normalize('NFD', text).encode('ascii', 'ignore')
                                             .decode("utf-8")
    str_text = str(text)
    str_text = str_text.replace('-', ' ')
    return str_text


headers = {}
headers["Accept"] = "application/json"


# The query bellow returns only 10 documents
response = requests.request(method='get', url=url,
                                        json={"query":{"match_all":{}}})
json_object = json.loads(response.content)

total_terms = json_object["hits"]["total"]["value"]
print("number of terms", total_terms)


# The query bellow returns all the documents
response = requests.request(method='get', url=url,
                      json={"query":{"match_all":{}}, "size":total_terms})
json_object = json.loads(response.content)

glossary_list = json_object["hits"]["hits"]


f = open(filename, "w")
```

```
# processamento conforme está implementado em glossary_analyzer
for entry in glossary_list:
    item = entry["_source"]["term"]
    term_tokens = item.split(';')

    for term in term_tokens:
        term_asciifolding = asciifolding(term)
        term_lowercase = term_asciifolding.lower()
        term_trim = term_lowercase.strip()
        term_one_whitespace = re.sub(' +', ' ',term_trim)
        print(term_one_whitespace)
        f.write(term_one_whitespace)
        f.write("\n")

f.close()
```

Figura 7. Script que gera um arquivo com a lista de termos presentes em um glossário indexado.

Uma vez criados os índices, conforme o mapeamento da Figura 2, e gerado esse arquivo de vocabulário, conforme Figura 7, é necessário utilizá-lo para a indexação da perguntas e respostas segundo o mapeamento da Figura 5, por meio do filtro “terms_on_file”.

Depois desse processo de indexação, a consulta a um determinado documento pode ser realizada de acordo com a consulta presente na primeira linha da Figura 8. Esta requisição GET do protocolo HTTP realiza uma consulta ao documento cujo identificador é “ilpf_001” no índice chamado “responde500”, que equivale à primeira pergunta do livro sobre integração lavoura-pecuária-floresta (ILPF). A resposta a essa consulta, exibida nas linhas seguintes, não retorna o valor para “glossary”, porque este campo não faz parte do documento original, conforme mostra a Figura 8.

```
GET responde500/_doc/ilpf_001
```


```
{
  "_index" : "responde500",
  "_type" : "_doc",
  "_id" : "ilpf_001",
  "_version" : 1,
  "_seq_no" : 500,
  "_primary_term" : 1,
  "found" : true,
  "_source" : {
    "question_number" : 1,
    "question" : "O que é integração lavoura-pecuária-floresta (ILPF)?",
    "answer" : "<p>É um sistema de produção sustentável que integra
atividades agrícolas, pecuárias e florestais, realizadas na mesma área, em
cultivo consorciado, em sucessão ou em rotação, e busca efeitos sinér-
gicos entre os componentes do agroecossistema, contemplando a adequação
ambiental, a valorização do homem e a viabilidade econômica da atividade
agropecuária.</p>",
    "chapter" : "Conceitos e Modalidades da Estratégia de Integração
Lavoura-Pecuária-Floresta",
    "book" : "Integração Lavoura-Pecuária-Floresta",
    "book_id" : "ilpf",
    "epub" : "https://mais500p500r.sct.embrapa.br/view/
epubs/90000033/90000033-ebook-epub.epub",
    "pdf" : "https://mais500p500r.sct.embrapa.br/view/pdfs/90000033-ebook-
-pdf.pdf",
    "year" : 2015
  }
}
```


Figura 8. Resposta à consulta ao documento ‘ilpf_001’ do índice ‘responde500’.

No entanto, é possível acessar os termos indexados em “glossary” por meio do recurso de “_termvectors”, uma vez que o parâmetro correspondente “term_vector” o habilita neste campo. Assim, uma consulta como mostra a primeira linha da Figura 9, explorando este recurso, possibilita a identificação dos termos do glossário presentes no documento. No resultado, mostrado nas linhas seguintes, alguns dados irrelevantes para o escopo deste documento são omitidos, mas é possível observar que ocorrem no documento “ilpf_001” do índice “responde500” os termos identificados por “agroecossistema”, “cultivo consorciado”, “integracao lavoura pecuaria” e “integracao lavoura pecuaria floresta”.

```
GET responde500/_termvectors/ilpf_001?fields=glossary
```

```
{
  "_index" : "responde500",
  "_id" : "ilpf_001",
  :
  "term_vectors" : {
    "glossary" : {
      "field_statistics" : {... },
      "terms" : {
        "agroecossistema" : {
          "term_freq" : 1
        },
        "cultivo consorciado" : {
          "term_freq" : 1
        },
        "integracao lavoura pecuaria" : {
          "term_freq" : 1
        },
        "integracao lavoura pecuaria floresta" : {
          "term_freq" : 1
        }
      }
    }
  }
}
```

Figura 9. Consulta aos termos do glossário presentes em um documento.

Uma vez identificados os termos, é possível obter suas definições por meio de consultas individuais a cada um dos termos conforme a Figura 10, ou por meio de uma consulta geral como na Figura 11. Esses passos precisam ser implementados pelo desenvolvedor que utiliza a API Responde Agro.

```
GET glossary/_search
{
  "query": {
    "match": {
      "term": "agroecossistema"
    }
  }
}
```

Figura 10. Consulta à definição de um único termo “agroecossistema”

```
GET glossary/_search
{
  "query": {
    "bool": {

      "should" : [
      {
        "query_string" : {
          "fields" : ["term"],
          "query" : "agroecossistema"
        }
      },
      {
        "query_string" : {
          "fields" : ["term"],
          "query" : "cultivo consorciado"
        }
      },
      {
        "query_string" : {
          "fields" : ["term"],
          "query" : "integracao lavoura pecuaria"
        }
      },
      {
        "query_string" : {
          "fields" : ["term"],
          "query" : "integracao lavoura pecuaria floresta"
        }
      }
      ]
    }
  }
}
```

Figura 11. Consulta a definições de vários termos simultaneamente.

Para a validação desta metodologia, foram usados glossários da Embrapa sobre ILPF (Telles et al., 2021), tema de um dos livros indexados, e sobre o Código Florestal (Embrapa, 2022a), que apresentam principalmente termos associados à produção agropecuária sustentável. Alguns desses termos são apresentados com siglas, como 'Zoneamento Agrícola de Risco Climático (ZARC)'. Em casos assim, dois termos, o completo e sua sigla, são especificados para uma mesma definição e separados por ';' conforme exemplificado na Figura 3.

Relações entre perguntas dos livros

Quando um usuário busca informações sobre determinado tema, é natural que procure encontrar vários documentos relacionados entre si. Assim, no caso do sistema de busca considerado neste trabalho, é esperado que um usuário deseje saber mais sobre determinado assunto consultando perguntas e respostas similares ao que ele faz originalmente. Uma maneira de auxiliá-lo nesta busca por mais informações relacionadas a determinado conteúdo é exibir perguntas e respostas parecidas com as que ele está visualizando no momento. Isso pode ser alcançado por um mecanismo que possibilite uma consulta por documentos, de maneira que uma busca é realizada em função de uma entrada constituída por um documento completo.

Segundo Yang et al. (2009), a consulta por documento - *Query by Document* (QBD) - permite ao usuário submeter um documento de texto como uma consulta e identificar documentos relacionados. Os autores propuseram técnicas para extrair frases do documento de entrada para gerar as consultas e buscar documentos similares em uma aplicação envolvendo textos de blogs.

É comum nesses trabalhos de QBD a construção de consultas baseadas em sentenças compostas por mais de uma palavra que estão presentes no documento de entrada, além da utilização de indicadores de similaridade baseados na frequência dos termos, os quais podem ser diretamente extraídos dos índices construídos para os mecanismos de busca.

Yang et al. (2009), por exemplo, usaram um mecanismo de ranqueamento baseado na informação de *term frequency - inverse document frequency* (TF-IDF) dos termos, assim como Lechtenberg et al. (2022) apresentaram um

método de QBD envolvendo textos científicos com abordagem que também explora TF-IDF. Marcos-Pablos e García-Peñalvo (2018) definem TF-IDF como um cálculo estatístico que considera tanto a ocorrência de um termo em um certo documento quanto sua cardinalidade no espaço de documentos, mas não leva em consideração a proximidade entre os termos no documento.

O BM25 é uma função utilizada na recuperação de informação baseada no TF-IDF e representa certo avanço em relação a este. O BM25 é a função de similaridade padrão do Elasticsearch, tecnologia empregada na implementação deste trabalho.

Pereira Júnior e Ziviani (2004) implementaram um processo para detectar e recuperar documentos similares, mas na web, o que configura um ambiente muito mais amplo do que o caso tratado no presente trabalho. Esse processo é composto por três etapas: a) a geração de uma “impressão digital” do documento, uma identificação única; b) a extração de sentenças dessa “impressão digital” para buscar candidatos a documentos similares; e c) a comparação do documento de entrada a cada candidato. Esses três passos também estão presentes na solução proposta por Dasdan et al. (2009).

Williams et al. (2014) apresentaram o SimSeerX, um mecanismo de busca para a recuperação de documentos similares que recebe documentos completos como consultas e retorna uma lista ranqueada de documentos similares. O SimSeerX é composto por interface de usuário e subsistemas de indexação e consulta. Sua principal diferença em relação aos outros sistemas, segundo os autores, é sua arquitetura genérica para QBD, como um *framework*, que possibilita o uso de qualquer função de similaridade de documentos, em vez de focar em métodos específicos de consulta e ranqueamento, conforme normalmente ocorre em outros trabalhos, como os de Zhang e Lu (2016) e de Chen et al. (2018).

Em essência, o presente trabalho apresenta características similares a estes mencionados:

a) trata-se de um recurso de QBD. A pergunta selecionada pelo usuário é usada como entrada para buscar perguntas relacionadas a ela;

b) são usadas técnicas para extrair termos compostos por duas ou mais palavras. No sistema implementado, foram considerados os termos compostos

que fazem parte do Agrovoc, um vocabulário controlado coordenado pela FAO (2022) – Organização das Nações Unidas para a Alimentação e a Agricultura - que envolve conceitos e terminologia nas áreas ligadas à agricultura, e está disponível para uso público;

c) o ranqueamento e a similaridade entre documentos são baseados em estatísticas dos índices, como TF-IDF e BM25. A própria tecnologia Elasticsearch empregada neste trabalho é baseada nessas funções para o cáculo de similaridade e para a geração do ranking dos resultados;

d) processo composto por etapas que envolvem a identificação de um documento, a extração de termos compostos para a busca de documentos candidatos e a comparação entre o documento de entrada e os candidatos. Neste trabalho, cada documento é, de certa maneira, representado por uma lista de termos, dentre os quais também termos compostos por duas palavras ou mais. Além disso, ocorre um ranqueamento comparando o documento de entrada a candidatos a documentos similares;

e) o sistema tem componentes para indexação, busca e interface de usuário. Estes componentes são detalhados adiante.

Além dessas características, há outras possibilidades para trabalhos futuros. Por exemplo, Oseledets et al. (2014) exploram a ideia de que dois objetos são similares se são referenciados por objetos similares. No presente trabalho, as perguntas são tratadas independentemente, embora possam ser agrupadas por livros e por capítulos de livros. É possível identificar relações entre diferentes perguntas, mas elas não são organizadas atualmente em estruturas que registram essas relações. Portanto, no contexto atual, não se pretende explorar essa ideia de que são similares documentos referenciados por documentos similares.

Outras técnicas podem ser exploradas. Por exemplo, experimentos de Chen et al. (2018) em uma aplicação para mapear da melhor maneira possível artigos científicos a conferências, mostram que o emprego de aprendizado ativo (*active learning*) apresenta melhores resultados do que o uso do BM25, mas requer anotação de exemplos. O aprendizado ativo é um processo interativo em que o sistema iterativamente mostra dados não anotados para o usuário, solicita marcação e a utiliza para o algoritmo de aprendizado, até que deter-

minada condição de parada é alcançada. Marcos-Pablos e García-Peñalvo (2018) também usaram uma metodologia iterativa que aplica mineração de textos e aprendizado de máquina. Neste trabalho, no entanto, a ideia é não explorar abordagens que envolvam trabalho humano adicional devido à limitação deste recurso.

A Figura 12 mostra os elementos complementares ao mapeamento que possibilitam a recuperação de documentos similares. Novos campos “related_questions” e “agrovoc” foram criados para auxiliar na representação de cada documento para a busca de documentos similares. Ambos envolvem conteúdo dos campos relacionados aos textos de pergunta, resposta, capítulo e título do livro, como pode ser observado pelo parâmetro “copy_to” destes campos. O campo “glossary” é definido conforme já apresentado na Figura 5. O parâmetro “term_vector” tem valor “no” para ambos os campos “related_questions” e “agrovoc” porque não há a previsão de recuperação desses termos, então não é necessário armazená-los, de maneira que aparecem apenas nos índices.

```
PUT responde500 {
  "settings": {
    "analysis": {
      "filter": {
        :
        "ptbr_stop_filter": {
          "type": "stop",
          "ignore_case": true,
          "stopwords": ["_brazilian_", "no"]
        },
        "terms_on_agrovoc": {
          "type":"keep",
          "keep_words_path":"agrovoc_vocabulary.txt"
        }
      },
      "analyzer": {
        :
        "stop_QA_analyzer": {
          "char_filter": ["html_strip"],
          "tokenizer": "uax_url_email",
          "filter": ["asciifolding", "lowercase", "trim", "ptbr_stop_filter"]
        },
        "agrovoc_analyzer":{
          "char_filter": ["html_strip"],
          "tokenizer": "uax_url_email",
          "filter": ["asciifolding", "lowercase", "trim", "shingle_term",
                     "terms_on_agrovoc", "ptbr_stop_filter"]
        }
      }
    }
  },
  "mappings": {
    "properties": {
      "question":{
        "copy_to": ["glossary", "related_questions", "agrovoc"],
        :
      },
      "answer":{
        "copy_to": ["glossary", "related_questions",  "agrovoc"],
        :
      },
      "chapter":{
        "copy_to": ["related_questions", "agrovoc"],
        :
      },
      "book": {
        "copy_to": ["related_questions", "agrovoc"],
        :
      },
      :,
      "related_questions":{
        "type": "text",
        "term_vector": "no",
        "analyzer": "stop_QA_analyzer"
```

```
      },
      "agrovoc": {
        "type": "text",
        "term_vector": "no",
        "analyzer": "agrovoc_analyzer"
      }
    }
  }
}
```

Figura 12. Elementos complementares no mapeamento que possibilitam a recuperação de documentos similares.

O "related_questions" utiliza o analisador "stop_QA_analyzer", enquanto o "agrovoc" utiliza "agrovoc_analyzer". O primeiro envolve o filtro de caracteres "html_strip", que remove *tags* HTML do texto, altém do *tokenizer* "uax_url_email" e filtros de tokens já considerados anteriormente, mas também inclui o "ptbr_stop_filter", que remove *stopwords*, palavras consideradas irrelevantes. O parâmetro "ignore_case" desse filtro determina que não ocorre diferenciação entre letras maiúsculas e minúsculas. O parâmetro "stopwords" determina que a lista de palavras irrelevantes é a padrão do Elasticsearch para o português brasileiro. Esta lista pode não conter algumas palavras que são importantes. Por exemplo, a palavra "no" não consta nesta lista, mas normalmente é considerada como *stopword*, então é incluída enumerando-a no mesmo parâmetro.

Essencialmente, então, o campo “related_questions” pode representar o documento como uma lista contendo seus termos já processados e ignorando *stopwords*. Por exemplo, o texto “O que é integração lavoura-pecuária-floresta (ILPF)?” gera a lista de tokens [“integracao”, “lavoura”, “pecuaria”, “floresta”, “ilpf”]. A ideia é representar um documento por suas principais palavras.

Já o campo “agrovoc” utiliza o analisador “agrovoc_analyzer”. Este é similar ao “glossary_doc_analyzer”, explicado anteriormente, exceto pelo conjunto de termos utilizado e pelo uso do filtro “ptbr_stop_filter”. Enquanto o “glossary_doc_analyzer” usa a lista de termos presentes no glossário indexado, o “agrovoc_analyzer” usa a lista de termos do Agrovoc. É importante utilizar o filtro “ptbr_stop_filter” em último lugar pelo seguinte motivo. Ele não pode ser aplicado antes do “shingle_term” porque muitos dos termos considerados utilizam *stopwords*, como, por exemplo, ‘de’ em ‘absorcao de agua’, e ‘no’ e ‘da’ em ‘mudanca no uso da terra’. Esses termos nunca seriam localizados se as *stopwords* fossem eliminadas antes do processamento dos termos compostos. Dependendo do vocabulário utilizado, esse filtro nem precisaria ser utilizado, assim como não é usado em “glossary_doc_analyzer”, mas o Agrovoc inclui termos que causam distorções no sistema se as *stopwords* não forem removidas. Por exemplo, os termos “o” e “se” aparecem no arquivo gerado a partir do Agrovoc porque “Se” é termo alternativo ao “Selênio” e “O” ao “Oxigênio”. Quando processados, resultam em “se” e “o”, considerados *stopwords* na língua portuguesa. O problema é que, na grande maioria das vezes, a ocorrência desses termos não se refere aos termos mencionados. Então, para anular seu efeito na representação dos documentos, esses termos são removidos pelo analisador com o uso do “ptbr_stop_filter”.

Uma vantagem do “agrovoc_analyzer” em relação ao “stop_QA_analyzer” é que consegue tratar casos em que trechos são frequentes, mas que não representam propriamente o conteúdo do documento. Por exemplo, uma sentença que contém o trecho “devem ser feitas” é relativamente frequente nos livros considerados, mas ao buscar perguntas similares, não é desejável que esse trecho seja considerado, já que não carrega significado relevante. No caso do analisador “agrovoc_analyzer”, esse trecho é ignorado porque não contém termos considerados pelo Agrovoc, mas o “stop_QA_analyzer” atribui certa relevância a suas palavras individualmente, uma vez que nenhuma delas é considerada *stopword*.

Assim como foi desenvolvido um script para gerar um arquivo com a lista de termos presentes em um glossário previamente indexado, exibido na Figura 7, um outro script foi construído para gerar o arquivo com os termos presentes no Agrovoc. Ele é apresentado em partes nas Figuras 13-16.

```
import requests
import json
import re
import time
import heapq

# Agrovoc URL: query the portughese terms with query ...
url_base = "https://agrovoc.uniroma2.it/agrovoc/rest/v1/
                                                search/?lang=pt&query="
filename = "agrovoc_vocabulary.txt"
```

Figura 13. Script que gera um arquivo com a lista de termos presentes no Agrovoc (parte 1): importação de pacotes e determinação de variáveis de entrada.

```
start_time = time.time()
start_partial_time = start_time
letters = [x for x in bytearray(range(97,123)).decode("utf-8")]
# ['a', 'b', ... 'z']

agrovoc_heap = []
heapq.heapify(agrovoc_heap) # Convert list to heap

# query initiating with letter a, b, ..., z
for letter in letters:
    query = letter + '*'  # 'a*', 'b*', .., 'z*'
    url = url_base + query
    print(url)

    response = requests.get(url)
    json_object = json.loads(response.content)
    term_list = json_object["results"]

    for term_entry in term_list:
        if "prefLabel" in term_entry:
            term = term_entry["prefLabel"]
            if term not in agrovoc_heap:
                heapq.heappush(agrovoc_heap, term)

        if "matchedPrefLabel" in term_entry:
            term = term_entry["matchedPrefLabel"]
            if term not in agrovoc_heap:
                heapq.heappush(agrovoc_heap, term)

        if "altLabel" in term_entry:
            term = term_entry["altLabel"]
            if term not in agrovoc_heap:
                heapq.heappush(agrovoc_heap, term)

    end_partial_time = time.time()
    partial_time = end_partial_time - start_partial_time
    print("Tempo: ", partial_time , "s")
    start_partial_time = end_partial_time
    print("Total of ", len(agrovoc_heap), " terms")

agrovoc_list = [heapq.heappop(agrovoc_heap)
                                        for i in range(len(agrovoc_heap))]
print("Size of the list:", len(agrovoc_list))

end_time = time.time()
total_time = end_time - start_time
print("Tempo de execução: ", total_time)
```

Figura 14. Script que gera um arquivo com a lista de termos presentes no Agrovoc (parte 2): leitura dos termos do Agrovoc.

```
# remove observations in parentesis after the term string and
# do the same text processing as GenerateVocabularyFromGlossary

# asciifolding from GenerateVocabularyFromGlossary
def asciifolding(text):
    import unicodedata
    try:
        text = unicode(text, 'utf-8')
    except NameError:
        pass
    text = unicodedata.normalize('NFD', text).encode('ascii', 'ignore')
                                             .decode("utf-8")
    str_text = str(text)
    str_text = str_text.replace('-', ' ')
    return str_text

agrovoc_terms = []
for term in agrovoc_list:

    # Some terms come with observation in parentesis. Eliminate them
    pattern = r'\(.*\)' # qualquer texto entre parênteses
    replacement = r''   # subsitui por vazio
    term1 = re.sub(pattern, replacement, term)

    # remove ',' from the term, as in '2,4,5 t' => '2 4 5 t'
    term2 = re.sub(r',', ' ', term1)

    # remove '.' from the term, as in
    # 'abies cilicica subsp. cilicica' => 'abies cilicica subsp cilicica'
    term3 = re.sub(r'\.', ' ', term2)

    # steps of GenerateVocabularyFromGlossary
    term_asciifolding = asciifolding(term3)
    term_lowercase = term_asciifolding.lower()
    term_trim = term_lowercase.strip()
    term_one_whitespace = re.sub(' +', ' ',term_trim)
    agrovoc_terms.append(term_one_whitespace)

# remove duplicated terms
unique_terms_agrovoc = []
for term in agrovoc_terms:
    if term not in unique_terms_agrovoc:
        unique_terms_agrovoc.append(term)

unique_terms_agrovoc.sort()
```

Figura 15. Script que gera um arquivo com a lista de termos presentes no Agrovoc (parte 3): processamento dos termos.

```
f = open(filename, "w")
for term in unique_terms_agrovoc:
    f.write(term)
    f.write("\n")
f.close()
```

Figura 16. Script que gera um arquivo com a lista de termos presentes no Agrovoc (parte 4): geração do arquivo.

O script começa com a importação de pacotes necessários e a definição de variáveis para a URL que permite o acesso ao Agrovoc e para o nome do arquivo a ser gerado com a lista de termos (Figura 13). Os termos são acessados conforme sua letra inicial por meio de um procedimento iterativo em que são realizadas consultas com entradas de 'a*' a 'z*'. É possível observar, para cada letra inicial, o tempo consumido, em segundos, e a quantidade de termos recuperados. Não apenas os termos principais são identificados, mas também os termos associados a eles, como os preferenciais e os alternativos (Figura 14).

Os termos, então, são processados (Figura 15) da mesma maneira como são processados os termos do glossário pelo script da Figura 7. No entanto, os termos do Agrovoc ainda passam pelos seguintes tratamentos adicionais: a) eliminação de observações entre parênteses; b) substituição de vírgulas por espaços em branco; e c) substituição de ponto final por string vazia. Assim, termos como "solo pantanoso (permanente)", "2,4,5 t" e "abies cilicica subsp. cilicica" são transformados em "solo pantanoso", "2 4 5 t" e "abies cilicica subsp cilicica". Dessa forma, observações relacionadas ao termo não impedem a detecção do próprio termo, e elementos como a vírgula e o ponto final, considerados separadores de tokens, não prejudicam a detecção dos termos que os contêm. Os duplicados são posteriormente removidos. Finalmente, os termos são escritos sequencialmente em arquivo, um por linha (Figura 16). O arquivo gerado é usado pelo filtro "terms_on_agrovoc" e recebeu o nome

de “agrovoc_vocabulary.txt” nas Figuras 12 e 13. Ele lista 41.533 termos. Em um computador padrão, a execução do script foi concluída em menos de dois minutos.

Depois da indexação, a consulta por documentos similares pode ser realizada explorando-se principalmente os campos “agrovoc” e “related_questions”. A Figura 17 exibe um *template* de busca a documentos similares cujo identificador é “related_one_book”.

```
PUT _scripts/related_one_book
{
  "script":{
    "lang":"mustache",
    "source":
    {
      "query" : {
        "bool": {
          "must": [
            { "match": { "book_id": "{{book_id}}" }}
          ],
          "should" : [
          {
            "query_string" : {
              "fields" : ["question^2", "answer", "related_questions^5",
                          "agrovoc^10"],
              "query" : "{{query_question}}",
              "boost": 2
            }
          },
          {
            "query_string" : {
              "fields" : ["question^1.2", "answer^0.6",
                          "related_questions^5", "agrovoc^10"],
              "query" : "{{query_answer}}",
              "boost": 1.2
            }
          }
          ]
        }
      }
    },
    "params": {
      "query_question": "",
      "query_answer": "",
      "book_id":""
    }
  }
}
```

Figura 17. *Template* de busca para perguntas relacionadas

Os *templates* de busca do Elasticsearch são buscas armazenadas que podem ser executadas com diferentes valores de parâmetros. Eles permitem a execução de buscas sem exposição da sintaxe de consultas da tecnologia e mudanças sem alteração de código do sistema que usa as APIs.

Esse *template* considera três parâmetros: “book_id”, que identifica o livro considerado, além de “query_question” e “query_answer”, textos da pergunta e da resposta do que está sendo considerado como documento de entrada na busca de documentos similares. A Figura 18 exibe um exemplo de chamada a esse *template*, em que se busca documentos similares àquele que apresenta pergunta e resposta conforme os parâmetros “query_question” e “query_answer” no livro sobre ILPF. O resultado equivale ao obtido por uma consulta normal ao mecanismo de busca, com os documentos contendo seus dados e listados em um ranking. Assim, é possível, por exemplo, extrair os textos das questões, para exibi-los em uma interface de usuário, ordenados de acordo com o ranking.

```
GET /responde500/_search/template
{
  “id”: “related_one_book”,
  “params”: {
    “query_question”: “O que é integração lavoura-pecuária-floresta
(ILPF)?”,
    “query_answer”: “<p>É um sistema de produção sustentável que
integra atividades agrícolas, pecuárias e florestais, realizadas na
mesma área, em cultivo consorciado, em sucessão ou em rotação, e busca
efeitos sinérgicos entre os componentes do agroecossistema, contem-
plando a adequação ambiental, a valorização do homem e a viabilidade
econômica da atividade agropecuária.</p>”,
    “book_id”:”ilpf”
  }
}
```

Figura 18. Consulta a perguntas com o *template* de busca “related_one_book”

Outro *template*, cujo identificador é “related_all”, muito similar a esse, também foi implementado para buscar documentos relacionados em todos os livros indexados. As únicas diferenças para o *template* da Figura 17 são as ausências dos parâmetros “book_id” e da condição “must” que restringe a busca ao livro determinado pelo “book_id”.

Embora os *templates* tenham sido implementados conforme a Figura 17, algumas alterações podem ser realizadas de acordo com os seguintes critérios:

a) quais campos do documento são usados como entrada? Nessa implementação, são usados apenas aqueles referentes às análises padrões das perguntas e das respostas, mas os textos de capítulo e de título do livro também podem ser usados, assim como variações das perguntas e respostas geradas a partir de diferentes analisadores. Por exemplo, aqueles que fazem stemização ou que mantêm as palavras exatamente como fornecidas. A inclusão dos textos de capítulos e títulos levaria a uma maior tendência de encontrar perguntas de um mesmo livro e de capítulos com temas similares;

b) como o conteúdo dos campos pode ser utilizado (parâmetro “query”)? Na Figura 17, o texto de pergunta está em subconsulta diferente da que explora o texto da resposta, e a pergunta apresenta peso maior do que o da resposta, devido aos valores do parâmetro “boost”. No entanto, o conteúdo dos campos poderia ser concatenado em uma única string para a produção de apenas uma subconsulta no atributo “should” de “bool”. Neste caso, não haveria distinção de relevância entre trechos da pergunta e da resposta, como há nesta implementação;

c) quais campos do índice são utilizados e com quais pesos (parâmetro “fields”)? Foram utilizados vários campos com pesos diferentes entre si. Os campos “related_questions” e “agrovoc” foram construídos especificamente para melhorar os resultados dessas buscas por documentos similares, já que representam os documentos por meio de termos significativos, o primeiro ignorando *stopwords* e o segundo considerando apenas termos relevantes do contexto agrícola. Na Figura 17, ainda são considerados os campos de “question” e “answer” com menor peso, pois podem ajudar a identificar documentos semelhantes. No entanto, outros campos do índice também poderiam ser utilizados, como o “glossary”, previamente explicado, e outros que indexam os termos de maneiras alternativas. Os pesos de cada um desses

campos também podem ser ajustados conforme a relevância que se deseja atribuir a eles.

Portanto, a Figura 17 apresenta uma possibilidade para a busca de perguntas relacionadas a uma determinada pergunta. No entanto, ela pode ser realizada de diversas maneiras. Um conjunto maior de testes e validação é necessário para ajustar os parâmetros da busca a fim de obter resultados cada vez melhores.

Uma vez que os resultados são retornados para essa busca, processamento adicional pode auxiliar na determinação dos documentos semelhantes. Por exemplo, cada documento do resultado é associado a um *score* que determina, de certa maneira, o grau de semelhança entre ele e o documento de entrada. Um valor mínimo para esse *score* pode ser estabelecido para que o documento correspondente seja de fato considerado similiar ao documento de entrada.

A estratégia adotada para a recuperação de documentos similares não requer a implementação de novos algoritmos ou de estruturas complementares aos índices já existentes. Tudo foi viabilizado apenas com a exploração dos recursos da tecnologia empregada na implementação do mecanismo de busca. Além disso, não requer trabalho adicional de anotação de dados, o que consumiria o trabalho manual de especialistas.

Apresentação do conteúdo

Uma das interfaces de usuário construídas para a validação da API Responde Agro foi implementada em Liferay Digital Experience Platform (DXP), em atendimento a uma demanda relacionada ao projeto Hubtech da Agricultura Familiar, que visa à disponibilização de informação e conteúdo agropecuário para extensionistas, agricultores e outros públicos relacionados (Brasil, 2022).

A Figura 19 exibe parte da página web que possibilita o uso do mecanismo de busca construído com base na API. O usuário pode começar a fazer uma consulta até encontrar uma pergunta no sistema que corresponda ao que deseja saber. Uma vez selecionada a opção com uma pergunta do livro, o

sistema exibe a resposta correspondente com informações adicionais, como o número da pergunta, o capítulo em que está e o ano de publicação do livro. Neste caso, o sistema está em um contexto de informações sobre a cultura do feijão-caupi. Então, todas as buscas consideram apenas o livro sobre esse tema.

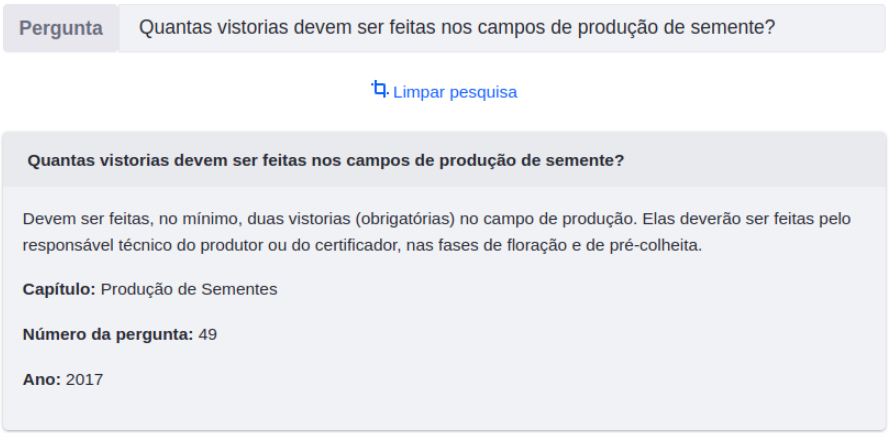


Figura 19. Interface de usuário para a API Responde Agro com um par de pergunta e resposta.

A tecnologia Liferay DXP utiliza portlets como componentes web para a construção de interfaces de usuário. Um portlet, segundo a sua especificação (Nicklous, 2016), é uma aplicação que oferece uma parte específica de conteúdo para compor uma página de portal. Os portlets são usados pelos portais como componentes de interface de usuário que oferecem uma camada de apresentação a sistemas de informação.

Bauer et al. (2015) usaram portlets para construir o WikiHyperGlossary, uma aplicação para documentos de química que provê a exibição de informações adicionais a documentos por meio de hiperlinks. Quando o usuário clica em um link associado a um termo, ele obtém seu significado de acordo com um glossário pré-determinado. Isso é feito com a exibição de uma janela sobreposta à página.

Vaz e Barbedo (2021) também implementaram uma solução com portlets para exibir um glossário relacionado a documentos técnico-científicos, de maneira que os termos presentes nos documentos listados como resultado de uma busca são definidos e exibidos em uma outra janela na mesma página. Esse glossário de termos é exibido por meio de um portlet dedicado que se comunica com os outros portlets da aplicação.

Este trabalho tem como foco a API Responde Agro e não interfaces de usuário que a utilizam, mas a Figura 20 mostra como uma interface pode ser implementada associando ao mecanismo de busca recursos para a exibição de glossários e de perguntas similares. Embora não tenha sido de fato implementada, essa representação de interface mostra como ela pode ser desenvolvida, seja com portlets ou outras tecnologias.

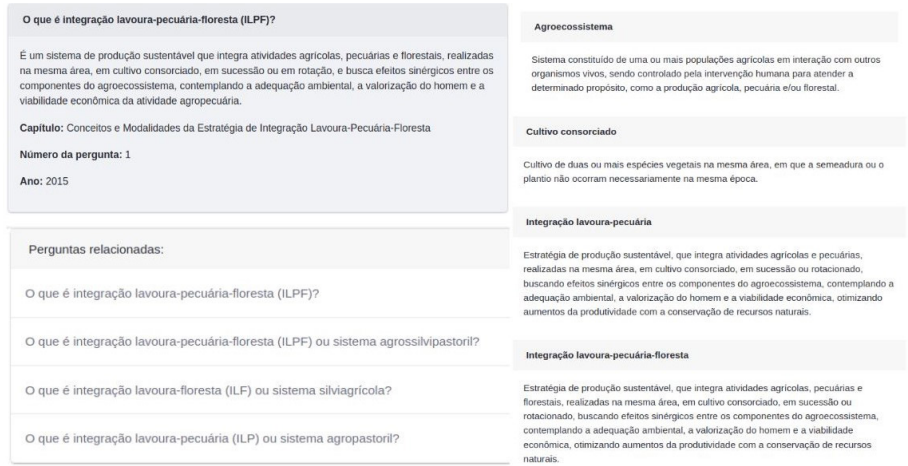


Figura 20. Esquema de interface de usuário para o mecanismo de busca com a exibição do resultado da consulta, dos termos do glossário no documento selecionado e de perguntas similares.

A Figura 21 mostra o fluxo de chamadas e seus resultados que possibilitam a exibição das informações da Figura 20. Uma vez que a pergunta foi selecionada, o código do documento é conhecido e isso possibilita a obtenção das informações referentes a essa pergunta com consultas diretas à API conforme a Figura 8. Seu resultado é suficiente para a exibição das informações do documento. Simultaneamente, os termos do glossário utilizado e que estão presentes na pergunta ou na resposta do documento selecionado podem ser

obtidos por uma chamada conforme a Figura 9. Com os dados do documento selecionado, é possível recuperar perguntas relacionadas a ele por meio de chamadas como a exibida na Figura 18. O resultado é uma lista de documentos com os campos relevantes para o recurso de perguntas similares. Da mesma maneira, com os termos do glossário já identificados no documento selecionado, é possível obter suas definições a partir de consultas como a apresentada na Figura 11. Ao fim desse fluxo, as informações obtidas são justamente aquelas que podem ser utilizadas na interface do usuário, como no exemplo da Figura 20.

Documento selecionado: pergunta 1 do livro de ILPF (ilpf_001)

(Figura 8)

```
GET /responde500/_doc/ilpf_001
```

↓

```
{"_source": {
 "question_number": 1,
 "question": "O que é integra-
ção lavoura-pecuária-floresta
(ILPF)?",
 "answer": "<p>É um sistema de
produção sustentável que integra
atividades agrícolas, pecuárias e
florestais, ...</p>",
 "chapter": "...",
 "book": "...",
 "book_id": "ilpf",
 "epub": "...",
 "pdf": "...",
 "year": 2015}
}
```

↓

(Figura 18)

```
GET /responde500/_search/template
{
 "id": "related_one_book",
 "params": {
  "query_question": "O que é in-
tegração lavoura-pecuária-floresta
(ILPF)?",
  "query_answer": "<p>É um
sistema de produção sustentável
que integra atividades agrícolas,
pecuárias e florestais, ...</p>",
  "book_id":"ilpf"
 }
}
```

↓

(Figura 9)

```
GET responde500/_doc/ilpf_001/
              _termvectors?fields=glossary
```

↓

```
{"term_vectors": {
 "glossary": {
 "terms": {
  "agroecossistema": {"term_freq":1},
  "cultivo consorciado": {"term freq":1},
  "integracao lavoura pecuaria":
                          {"term_freq":1},
  "integracao lavoura pecuaria floresta":
                          {"term_freq":1},
   }
  }
 }
}
```

↓

(Figura 11)

```
GET glossary/_search
{"query": {
  "bool": {
   "should" : [
   {"query_string" : {
     "fields" : ["term"],
     "query" : "agroecossistema"}},
   {"query_string" : {
     "fields" : ["term"],
     "query" : "cultivo consorciado"}},
   {"query_string" : {
     "fields" : ["term"],
     "query":"integracao lavoura pecua-
ria"}},
   {"query_string" : {
     "fields" : ["term"],
     "query" : "integracao lavoura pecua-
ria floresta"}}
   ]
  }
 }
}
```

↓

```
{"hits" : {
  "max_score" : 201.19788,
  "hits" : [
  {"_id" : "ilpf_001",
   "_score" : 201.19788,
   "_source" : {
    "question" : "O que é inte-
gração lavoura-pecuária-floresta
(ILPF)?",
    :
   }
  },
  {"_id" : "ilpf_006",
   "_score" : 107.092674,
   "_source" : {
    "question" : "O que é inte-
gração lavoura-pecuária-floresta
(ILPF) ou sistema agrossilvipas-
toril?",
    :
   }
  },
  {"_id" : "ilpf_005",
   "_score" : 86.42349,
   "_source" : {
    "question" : "O que é inte-
gração lavoura-floresta (ILF) ou
sistema silviagrícola?",
    :
   }
  },
  {"_id" : "ilpf_003",
   "_score" : 83.435356,
   "_source" : {
    "question" : "O que é inte-
gração lavoura-pecuária (ILP) ou
sistema agropastoril?",
    :
   }
  },
  :
  ]
  }
}
```

```
{"hits" : {
  "hits" : [
  {"_index" : "glossary",
   "_source" : {...,
    "term": "Agroecossistema",
    "definition": "Sistema constituído de
uma ou mais populações agrícolas ...",
    "glossary_title" : "ILPF"}},
  {"_index" : "glossary",
   "_source" : {...,
    "term": "Cultivo consorciado",
    "definition": "Cultivo de duas ou mais
espécies vegetais na mesma área...",
    "glossary_title" : "ILPF"}},
  {"_index" : "glossary",
   "_source" : {...,
    "term": "Integração lavoura-pecuária",
    "definition": "Estratégia de produção
sustentável, que integra atividades
agrícolas e pecuárias ...",
    "glossary_title" : "ILPF"}},
  {"_index" : "glossary",
   "_source" : {...,
    "term": "Integração lavoura-pecuária-
-floresta",
    "definition": "Estratégia de produção
sustentável, que integra atividades
agrícolas, pecuárias e florestais, ...",
    "glossary_title" : "ILPF"}}
  ]
  }
}
```

Figura 21. Fluxo de chamadas à API para o uso dos recursos de glossário e de perguntas relacionadas

Discussão

Os experimentos relatados neste documento constituem protótipos e foram realizados em ambiente local de teste, conforme previsto no projeto EmbrapaIZagroSearch. Os recursos desenvolvidos ainda não foram incluídos na API Responde Agro, que já está disponível na plataforma AgroAPI para acesso restrito a instituições autorizadas. Novos testes devem ser conduzidos para se avaliar a inclusão na API dos recursos apresentados neste documento.

Para o caso do uso de glossários, é importante avaliar quais glossários podem trazer uma melhor experiência ao usuário de sistemas desenvolvidos com o uso da API Responde Agro, com informações mais detalhadas e preparadas por profissionais dos temas envolvidos, desde que considerado o contexto em que os termos são normalmente empregados.

Alguns termos podem ser usados com significados distintos em outros contextos. Por exemplo, a sigla URT, presente no glossário sobre ILPF e ligado a “Unidade de Referência Tecnológica”, pode não fazer sentido em outros contextos. Área abandonada é definida como “espaço de produção convertido para o uso alternativo do solo sem nenhuma exploração produtiva há, pelo menos, trinta e seis meses e não formalmente caracterizado como área de pousio”. Isso faz sentido quando se considera a produção agropecuária, mas pode ter outro significado em outros contextos.

Uma vantagem é que o uso dos recursos de glossário e de perguntas relacionadas não é obrigatório. O usuário da API pode escolher usar estes recursos ou simplesmente ignorá-los. Eles foram desenvolvidos independentemente, mas podem ser agregados com facilidade, já que as alterações introduzidas para um recurso não afetam o outro.

Conclusões

Foram estudados e desenvolvidos neste trabalho recursos que podem melhorar a experiência de uso de mecanismos de busca, mais especificamente aqueles construídos com base na API Responde Agro, que permite acesso ao conteúdo da coleção “500 Perguntas 500 Respostas”, da Embrapa.

Os testes realizados com o protótipo desenvolvido permitem concluir que o recurso de glossário pode fazer parte da API Responde Agro e ser explorado por interfaces de usuário que a utilizam. O único trabalho adicional é a seleção de glossários que promovam uma melhor experiência para uso no contexto da API e sua preparação no formato adequado para a inclusão nos índices.

Já o recurso de perguntas similares, embora tenha apresentado bons resultados, ainda requer um maior estudo e avaliação das possíveis configurações de parâmetros a fim de se obter melhores resultados na busca por documentos similares.

Em relação à apresentação desses recursos, o presente documento limita-se a apresentar a forma como a API Responde Agro pode ser usada para se construir uma interface de usuário que explore os recursos de perguntas similares e de glossário.

Os recursos desenvolvidos, portanto, mostraram-se promissores para tornar mais ricas as interfaces de usuário para a recuperação da informação das obras consideradas, possibilitando a oferta de informações complementares que são relevantes, confiáveis e contextualizadas.

Referências

BAUER, M. A.; BERLEANT, D.; CORNELL, A. P.; BELFORD, R. E. WikiHyperGlossary (WHG): an information literacy technology for chemistry documents. **Journal of Cheminformatics**, v. 7, 22, 2015. DOI: 10.1186/s13321-015-0073-7.

BRASIL. Ministério da Agricultura, Pecuária e Abastecimento. Secretaria de Agricultura Familiar e Cooperativismo. Portaria SAF/MAPA n° 287, de 16 agosto de 2022. Institui o Projeto Hubtech da Agricultura Familiar, o qual trata do desenvolvimento de arranjos institucionais - Hubs Virtuais para disponibilizar informação e conteúdo agropecuário relevante para os extensionistas, agricultores e outros públicos relacionados, convergindo as ações de diversas instituições do Agro Brasileiro. **Diário Oficial da União**, 17 ago. 2022. Seção I, p. 11-18.

CHEN, S.; MOH´D, A.; NOURASHRAFEDDIN, S.; MILIOS, E. Active high-recall information retrieval from domain-specific text corpora based on query documents. In: ACM SYMPOSIUM ON DOCUMENT ENGINEERING, 18., 2018, Halifax. **Proceedings…** New York: Association for Computing Machinery, 2018. p. 1-10. DOI: 10.1145/3209280.3209532.

DASDAN, A.; D´ALBERTO, P.; KOLAY, S.; DROME, C. Automatic retrieval of similar content using search engine query interface. In: ACM CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT, 18., Hong Kong. **Proceedings…** New York: Association for Computing Machinery, 2009. p. 701-710. DOI: 10.1145/1645953.1646043.

ELASTIC. **Elasticsearch guide**. Disponível em: https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html. Acesso em: 29 ago. 2022.

EMBRAPA. **Código Florestal**: adequação da paisagem rural: glossário. Disponível em: https://www.embrapa.br/en/codigo-florestal/entenda-o-codigo-florestal/glossario. Acesso em: 22 ago. 2022a.

EMBRAPA. **Coleção 500 Perguntas 500 Respostas:** você pergunta, a Embrapa responde. Disponível em: https://mais500p500r.sct.embrapa.br/view/index.php. Acesso em: 2 ago. 2022b.

FAO. **AGROVOC**. Disponível em: https://www.fao.org/agrovoc/. Acesso em: 25 ago. 2022.

LECHTENBERG, F.; FARRERES, J.; GALVAN-CARA, A. L.; SOMOZA-TORNOS, A.; ESPUÑA, A.; GRAELLS, M. Information retrieval from scientific abstract and citation databases: a query-by-documents approach based on Monte-Carlo sampling. **Expert Systems with Applications:** an International Journal, v. 199, 116967, 2022. Issue C. DOI: 10.1016/j.eswa.2022.116967.

MARCOS-PABLOS, S.; GARCÍA-PEÑALVO, F. J. Information retrieval methodology for aiding scientific database search. **Soft Computing**, v. 24, n. 8, p. 5551-5560, 2020. DOI: 10.1007/s00500-018-3568-0.

NICKLOUS, M. S. **Java™ Portlet Specification:** final: version 3.0. [S.l.]: IBM, 2016. Disponível em: https://download.oracle.com/otndocs/jcp/portlet-3-final-spec/. Acesso em: 31 out. 2022.

OSELEDETS, I. V.; OVCHINNIKOV, G. V.; KATRUTSA, A. M. Fast, memory-efficient low-rank approximation of SimRank. **Journal of Complex Networks**, v. 5, n. 1, p. 111-126, 2017. DOI: 10.1093/comnet/cnw008.

PEREIRA JÚNIOR, A. R.; ZIVIANI, N. Retrieving similar documents from the web. **Journal of Web Engineering**, v. 2, n. 4, p. 247-261, 2004.

TELLES, M. A.; PIEROZZI JUNIOR, I.; COIMBRA, E. C.; CORADINI, M. C.; TURCI, P. H.; ALENCAR, M. de C. F.; RASCHE, F. (ed.). **Glossário ILPF:** Integração Lavoura-Pecuária-Floresta = ICLF Glossary: Integrated Crop-Livestock-Forestry = Glosario ILPF: Integración Agricultura-Ganadería-Bosque. Colombo: Embrapa Florestas, 2021. 82 p. il. color. (Embrapa Florestas. Documentos, 350). Disponível em: https://ainfo.cnptia.embrapa.br/digital/bitstream/item/223001/1/Livro-Doc-350-1928-final-4.pdf. Acesso em: 28 out. 2022.

VAZ, G. J. **Analisadores e mapeamento de um Sistema de Recuperação de Informação (SRI) de publicações técnico-científicas agropecuárias.** Campinas: Embrapa Informática Agropecuária, 2017. 39 p. il. (Embrapa informática Agropecuária. Documentos, 152). Disponível em: https://www.infoteca.cnptia.embrapa.br/infoteca/handle/doc/1131603. Acesso em: 28 out. 2022.

VAZ, G. J.; APOLINÁRIO, D. R. de F.; CORREA, J. L.; VACARI, I.; GONZALES, L. E.; DRUCKER, D. P.; BARIANI, J. M.; EVANGELISTA, S. R. M.; ROMANI, L. A. S. AgroAPI: criação de valor para a Agricultura Digital por meio de APIs. In: CONGRESSO BRASILEIRO DE AGROINFORMÁTICA, 11., 2017, Campinas. **Ciência de dados na era da agricultura digital:** anais. Campinas: Editora da Unicamp: Embrapa Informática Agropecuária, 2017a. p. 59-68. SBIAgro 2017. Disponível em: https://www.alice.cnptia.embrapa.br/alice/handle/doc/1083275. Acesso em: 28 out. 2022.

VAZ, G. J.; BARBEDO, J. G. A. An information retrieval system based on multiple portlets:communication between its components. **International Journal of Web Portals**, v. 13, n. 1, p. 74-86, Jan./June 2021. DOI: 10.4018/IJWP.2021010105.

VAZ, G. J.; OLIVEIRA, L. H. M. de; PIEROZZI JÚNIOR., I. Visualização de glossário em sistemas de recuperação de informação. In: BRAZILIAN SYMPOSIUM IN INFORMATION AND HUMAN LANGUAGE TECHNOLOGY AND COLLOCATED EVENTS, 11., 2017, Uberlândia. **Proceedings of the conference**. Uberlândia: Sociedade Brasileira de Computação, 2017b. p. 83-92. STIL 2017. Disponível em: https://www.alice.cnptia.embrapa.br/alice/handle/doc/1077521. Acesso em: 28 out. 2022.

WILLIAMS, K.; WU, J.; GILES, C. L. SimSeerX: a similar document search engine. In: ACM SYMPOSIUM ON DOCUMENT ENGINEERING, 14., 2014, Fort Collins. **Advancing computing as a science & profession:** proceedings. New York: Association for Computing Machinery, 2014. p. 143-146. DOI: 10.1145/2644866.2644895.

YANG, Y.; BANSAL, N.; DAKKA, W.; IPEIROTIS, P.; KOUDAS, N.; PAPADIAS, D. Query by document. In: ACM INTERNATIONAL CONFERENCE ON WEB SEARCH AND DATA MINING, 2., 2009, Barcelona. **Proceedings…** New York: Association for Computing Machinery, 2009. p. 34-43. DOI: 10.1145/2644866.2644895.

ZHANG, Y.; LU, J. Near-duplicated documents in CiteSeerX. In: WORKSHOP ON SCHOLARLY BIG DATA, 2016, New York. **AI perspectives, challenges, and ideas:** proceedings. [S.l.: s.n., 2016]. p. 22-28.

Embrapa

Agricultura Digital

MINISTÉRIO DA AGRICULTURA, PECUÁRIA E ABASTECIMENTO

PÁTRIA AMADA BRASIL GOVERNO FEDERAL