- Home
- Exercícios
- Trabalho final

Jailson Nunes Leocadio

jailsonleocadio@usp.br | jailsonleocadio@gmail.com

Doutorando em Engenharia de Computação pela Escola Politécnica da Universidade de São Paulo.

Laboratório de Automação Agrícola
Núcleo de Pesquisa em Biodiversidade e Computação

Currículo Lattes: http://lattes.cnpq.br/6225687160783333

Estudo e pesquiso:

✔ *Citizen Science*;
✔ Ciência de Dados (*Data Mining*, *Knowledge Discovery in Database*, *AI*, *Machine Learning*);
✔ Algoritmos para construção de modelos de distribuição geográfica de espécies.

Exercícios

Veja abas no topo da página 😀.

Propostas de Trabalho Final

Primeira proposta: Acesso à base de Dados do iNaturalist

Contextualização

Os dispositivos móveis trouxeram o benefício da produção de dados em larga escala. Hoje em dia, eles possuem inúmeros sensores capazes de coletar dados automaticamente: acelerômetro, giroscópio, magnetômetro, GPS, barômetro, entre outros. Além disso, também recebem diversas informações através da rede da operadora de telefonia móvel ou pela conexão à internet. Dessa forma, eles se tornaram importantes aliados na coleta de dados para o desenvolvimento da ciência. Em pesquisas biológicas é comum o desenvolvimento de projetos que buscam incluir a população em atividades de monitoramento (ou outra etapa do trabalho); prática denominada *Citizen Science* (Ciência Cidadã).

Nesse contexto, o portal iNaturalist é uma base de dados global que armazena registros de presença de espécies biológicas, coletados por pessoas comuns através de uma interface web ou de um aplicativo móvel. Uma determinada observação adicionada à base de dados pode ser acessada por qualquer usuário cadastrado, que também pode sugerir uma identificação para o espécime registrado, concordar ou discordar do táxon atual, dentre outras interações. A qualidade de uma observação sobe de nível quando mais usuários concordam com as informações adicionadas.

O algoritmo

O algoritmo proposto fará conexão com a base de dados do *iNaturalist*, através de sua API pública, para retornar ao usuário um conjunto de dados de registros de espécimes, de acordo com os filtros passados pelos parâmetros.

Exemplo: `function("trichechus manatus", "Brasil", geo=TRUE)`

Essa chamada deve invocar a função, solicitando observações de *Trichechus manatus* registradas no Brasil e com dados de latitude e longitude.

Argumentos

Em negrito, os valores *default*. Nenhum é obrigatório, mas pelo menos um deve ser informado.

- `taxon_name`: *String* de caracteres - Informa qual o táxon das observações se deseja buscar;
- `place`: *String* de caracteres - Indica qual localização se deve restringir a busca. Serão aceitos apenas os locais cadastrados no site, ou seja, não há garantia que todos os municípios, estados e etc. sejam encontrados;
- `captive`: True ou **False** - Se observações de espécimes cativas ou capturadas devem ser consideradas;
- `geo`: True ou **False** - Se deve ser retornadas apenas observações com dados de geolocalização;
- `observed_on`: Date - Restringe as observações para a data informada;
- `quality_grade`: Factor (casual, needs id, **research**) - Restringe as observações para o fator escolhido;
- `date`: list(day, month, year) - Restringir a busca pra um dia, mês ou ano específico. Pode ser

informado apenas o dia, mês ou ano, dois deles ou todos;
- `circle`: list(lat, lng, radius) - Restringe a busca para uma dada circunferência;
- `maxresults`: (**500**) Número inteiro maior que zero - Informa o número máximo de observações que se deseja

Pseudo-código

1. Verifica quais argumentos foram recebidos pela função e valida a tipagem e limite dos dados;
2. Monta URL de busca de acordo com os argumentos recebidos;
3. Caso o usuário informe um local (*place*): acessar base de dados para tentar encontrar um local que case com o informado. Caso sim, salvar o ID para a busca das observações, caso não, informar ao usuário e considerar o local como não informado;
4. Realiza busca na base de dados através da URL;
5. Recebe os dados da base de dados em formatação JSON e converte para DATA TABLE;
6. Percorre os dados recebido, limpa e exclui os atributos que não serão mostrados pro usuário;
7. Plota todas as observações georreferenciadas em um mapa.

Retorno

- Observações em um DATA TABLE;
- Mapa com as observações georreferenciadas.

Segunda proposta: Recuperar informações de publicações científicas

Contextualização

Organizar as referências bibliográficas de um referencial teórico facilita a escrita do trabalho, evita o plágio e ajuda no desenvolvimento de mapeamentos sistemáticos, um tipo de levantamento bibliográfico que segue estritamente um protocolo de pesquisa. Atualmente, uma das ferramentas mais utilizadas para a edição de documentos acadêmicos é o *LaTex* e o arquivo que armazena a bibliografia nessa plataforma é um arquivo com extensão *.bib*. Para cada referência citada, o arquivo armazena o título, autores, ano de publicação, periódico, DOI, entre outros metadados, de maneira estruturada e com formatação específica.

Nem sempre dispomos de todas as informações sobre os artigos utilizados na pesquisa e a recuperação desses dados por meio de pesquisas individuais nos repositórios *online* demandaria bastante tempo. Os aplicativos gerenciadores de referências bibliográficas, Mendeley e Zotero, por exemplo, ajudam nesse propósito, porém não facilitam na exportação personalizada dos dados para um determinado tipo de arquivo escolhido pelo usuário e na análise exploratória dos dados.

O algoritmo

O algoritmo proposto receberá do usuário um arquivo *.bib* e acessará as bases de dados online do Web of Science e Scopus, através de suas API's, para completar os metadados faltantes.

Argumentos

- `file`: Dados formatado no estilo de arquivos *.bib*.

Pseudo-código

1. Verifica o argumento recebido pela função e faz sua validação;
2. Percorre os dados do arquivo e insere cada item do arquivo em um DATA TABLE (conterá algumas colunas pré-definidas, como: *título, autores, abstract, ano de publicação, id, área de concentração, país de origem, instituição afiliada, periódico, volume, edição, páginas*);
3. Percorre cada linha do DATA TABLE para verificar dados faltantes. Em caso de dados faltantes, será feita consulta nas bases de dados dos dois repositórios online para tentar encontrar o artigo e seus metadados. Em caso de sucesso, completar os dados; em caso negativo, manter o estado atual;
4. Gerar arquivo formatado no estilo *.bib* com os dados atualizados dos artigos;
5. Realizar plots exploratórios dos dados.

Retorno

- Trabalhos científicos atualizados em um DATA TABLE;
- Trabalhos científicos atualizados na formatação do arquivo *.bib*.
- Plots: histograma e/ou gráficos de pizza dos anos de publicação dos artigos, áreas de concentração, países de origem, afiliação, entre outros.

> Jailson, seu plano A parece super interesante e considero que oferecerá um nível ótimo de desafio para você. Sugiro fazer o plano A, embora considero super útil e também interessante a sua proposta de plano B.
>
> . — *SolimaryGarcia 2018/05/11 09:40*

> Obrigado! Ficarei com o plano A. — *Jailson Nunes Leocadio 2018/05/11 10:12*

Trabalho Final

Veja abas no topo da página 😀.

From:
http://ecor.ib.usp.br/ - **ecoR**

Permanent link:
http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2018:alunos:trabalho_final:jailsonleocadio:start

Last update: **2020/09/23 17:12**