Identifying endangered species in suspect mixtures: The *CITES-checker* pipeline

Youri Lammers[1], Rutger A. Vos[1], Thomas Bolderink[2], Alex Hoogkamer[2], Roeben Vink[2], Barbara Gravendeel[1,2]

Abstract

Mixtures of organic substances traded internationally frequently raise the suspicion that they contain materials obtained from species protected by the Convention on International Trade in Endangered Species of Wild Fauna and Flora (CITES). Examples of these include mixtures such as incense and traditional Chinese medicine. High-throughput sequencing of DNA barcode markers obtained from such samples may provide insight into their composition, but manual verification of the results against the CITES appendices is labor intensive. Here we present an analysis pipeline that automates this procedure. The *CITES-checker* pipeline is designed to process a set of (next generation) sequences to determine whether it contains genetic material obtained from species listed in the CITES appendices.

Keywords

Species identification; high-throughput sequencing; analysis pipeline; CITES

Introduction

The Convention on International Trade in Endangered Species of Wild Fauna and Flora (CITES) was signed in 1973 to control and regulate trade in endangered species. The convention produces lists ("appendices") of species in which trade is tightly controlled or prohibited under strict legal sanction. Materials obtained from species listed in CITES appendices, such as endangered orchids, are often used in herbal mixtures such as traditional Chinese medicines (TCMs). During the manufacture of such mixtures the biological contents, plant materials, are processed (ground, cooked, dried, blended with other products), which can make identification based on chromatographic methods difficult (Coghlan et al., 2012). DNA barcoding (Savolainen, Cowan, Vogler, Roderick, & Lane, 2005) is a powerful tool whereby the composition of such mixtures can be retrieved by sequencing a variable marker of the genome and comparing it against a reference database (Hebert, Cywinska, Ball, & deWaard, 2003). With high-throughput sequencing (HTS) techniques (Shendure & Ji, 2008) a large number of barcode sequences can be generated and analysed, which leads to a greater identifying potential for complex samples. The process of going through a set of identified sequences and manually comparing them to the CITES appendices is labor intensive, especially considering the increase in data produced through HTS. Here, we present a pipeline that automates both the

[1] Naturalis Biodiversity Center, Postbus 9517, 2300 RA, Leiden, the Netherlands

[2] University of Applied Sciences Leiden, Postbus 382, 2300 AJ, Leiden, the Netherlands

identification and CITES listing verification step to scan large numbers of both samples and sequences efficiently for the presence of genetic material from protected species.

Pipeline design

The pipeline verifies whether a sequence originates from a CITES protected species by comparing it to the NCBI GenBank's reference databases (Benson, Karsch-Mizrachi, Lipman, Ostell, & Sayers, 2009) using BLAST (Altschul, Gish, Miller, Myers, & Lipman, 1990) searches. The NCBI taxonomic identifier (taxon ID) of the resulting BLAST hits are compared to a list of taxon IDs that correspond to CITES-listed species. Any putative matches are reported back to the user including the immediately surrounding context of the CITES appendix text. The steps of the pipeline are shown in Figure 1.

Local CITES database

As an offline process, a local database is maintained containing the corresponding taxon IDs of CITES listed species. To update this database, a copy of the CITES appendix can be downloaded (Fig. 1 step 1) which the pipeline scans to retrieve names of CITES protected species and the appendix that contains them.

For each entry in the CITES appendix the corresponding taxon ID is retrieved using approximate string searches in the NCBI taxonomic database (Fig. 1 step 2). Since an entire genus or higher taxon can be listed in the CITES appendix (for example: Orchidaceae), higher taxa are expanded into all lower ranks to which GenBank sequences may be annotated. When no taxon ID can be obtained, a taxonomic name reconciliation web service (TNRS, http://api.phylotastic.org/tnrs, (Stoltzfus et al., n.d.)) is used (Fig. 1 step 3) to obtain a list of synonyms, based on which the pipeline retries to obtain taxon IDs. The taxon IDs for CITES protected species are locally stored, along with CITES appendix information and NCBI taxon names, in a comma separated value (CSV) file that can be read by standard spreadsheet software.

Sequence identification

To identify putative CITES-listed species from sequence data, the pipeline takes a set of sequences (in FASTA format) and searches these using BLAST against an NCBI GenBank database (Fig 1. 4) (all GenBank databases and BLAST algorithms are supported, by default the nucleotide database *nr* is used in combination with the *blastn* algorithm). Large sets of sequences are preferably clustered first into Operational Clustered Taxonomic Units (OCTUs), to reduce the number of redundant sequences.

As NCBI GenBank contains erroneous taxonomic annotations (Groenenberg, Neubert, & Gittenberger, 2011), the pipeline uses a user-editable blacklist of GenBank accession numbers for which taxonomic determination is known to be incorrect. For each BLAST hit that passes filtering through this blacklist (Fig 1. 5) the taxon ID is obtained from the sequence record. This taxon ID is then matched against the local CITES database (Fig 1. 6) to determine if the sequence originates from a listed species.

Pipeline output

The final result is a CSV spreadsheet file containing the query sequences, BLAST hits and, in case it is a CITES listed species, the surrounding textual context from the applicable appendix. A condensed example of such output is shown in Table 1.

CITES appendices contain multiple exceptions for certain species, e.g. based on their geographic location, domestication status or the enforcement of trade quota. The pipeline is not capable of handling these various exceptions, as they are not made available in a structured format. All results that match the names listed in the CITES appendices are therefore flagged and the surrounding context is reported to the user to determine if the putative hit is genuine or a false positive.

Usage examples

In its simplest form, the pipeline is run an on input FASTA file, producing an output CSV file, like so:

```
CITES_Check.py -i <infile> -o <outfile>
```

A user-specified blacklist file (which is a text file containing one GenBank accession number on each line) can be specified like so:

```
CITES_Check.py -i <infile> -o <outfile> -b <blacklist>
```

An alternate local database of reconciled CITES names can be specified like so:

```
CITES_Check.py -i <infile> -o <outfile> -c <database>
```

To force or avoid updating the local database, add `--force_update` or `--avoid_update`, respectively. In addition, numerous command line arguments to control BLAST search behavior can be provided. Sensible defaults have been defined, and their semantics can be recalled by issuing the `--help` argument.

Future work

Although the pipeline presented here is immediately useful, several modifications are possible that would increase usability and impact. For example, although incorrect taxonomic annotations of GenBank records have previously been noted (Groenenberg et al., 2011), no community project exists to record and track such errors (Pennisi, 2008). We note that the blacklist used by the CITES-checker could be used for such record keeping, especially as the infrastructure for this already exists by our usage of *git* as a decentralized revision control system. Conversely, should such a community wide blacklist exist (or come into existence), CITES-checker could be modified to make use of it. Another possible modification is the addition of a web-based graphical user interface, which would make the pipeline accessible to non-expert users as it would remove the need for local installations. In addition, this web application could be configured to update the local database of reconciled names and the blacklist at frequent intervals, thereby guaranteeing that the user always operates on the most recent knowledge.

Implementation

CITES-checker is written in python and uses the *bio-python*, *beautiful-soup* and *requests* packages to handle FASTA sequences and communicate with the various APIs and web services used.

Availability and requirements

- Project name: CITES-checker
- Project home page: https://github.com/ncbnaturalis/CITES-checker
- Operating system(s): Platform independent
- Programming language: Python (version 2.7 or 3.0 and higher)
- Other requirements: the non-core Python packages *bio-python*, *beautiful-soup*, *requests*
- License: BSD-3
- No restrictions to use by non-academics

List of abbreviations

- **API:** Application Programming Interface
- **BLAST**: Basic Local Alignment Search Tool
- **CITES**: Convention on International Trade in Endangered Species of Wild Fauna and Flora
- **CSV**: Comma Separated Values
- **NCBI**: National Center for Biotechnology Information
- **TCM**: Traditional Chinese Medicine
- **TNRS**: Taxonomic Name Reconciliation Service

Acknowledgements

The authors would like to thank Denise Pool for kindly providing us with data on which to test the pipeline and Sigrid Beiboer for co-mentoring TB, AH and RV during the internship out of which this project evolved.

Authors’ contributions

TB, AH and RV conceived of and developed a first prototype of *CITES-checker*, which YL re-implemented. YL and RAV contributed equally to the drafting of this manuscript. RAV oversaw software engineering, BG provided use cases, project management and additional writing. All authors have reviewed and approved the final version of this manuscript.

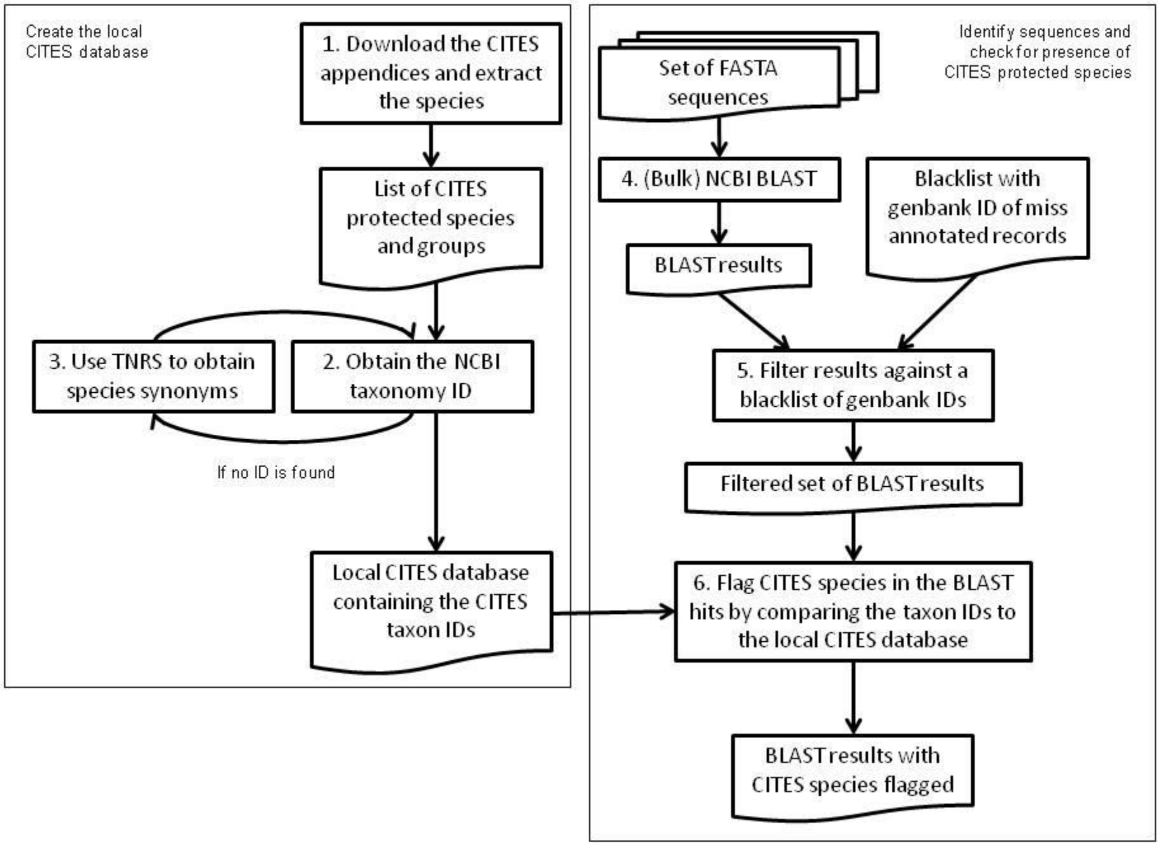
Competing Interests

The authors declare that they have no competing interests.

Figures

Figure 1: Steps of the analysis pipeline

The panel on the left shows the offline process of updating and reconciling names of taxa listed in CITES appendices with the NCBI taxonomy, the panel on the right shows the process of species identification from sequence data by matching against the reconciled database of CITES names. See text for details.


Create the local CITES database
1. Download the CITES appendices and extract the species
List of CITES protected species and groups
3. Use TNRS to obtain species synonyms
2. Obtain the NCBI taxonomy ID
If no ID is found
Local CITES database containing the CITES taxon IDs
Identify sequences and check for presence of CITES protected species
Set of FASTA sequences
4. (Bulk) NCBI BLAST
Blacklist with genbank ID of miss annotated records
BLAST results
5. Filter results against a blacklist of genbank IDs
Filtered set of BLAST results
6. Flag CITES species in the BLAST hits by comparing the taxon IDs to the local CITES database
BLAST results with CITES species flagged

Tables

Table 1: Example output

Condensed version of results obtained by running the pipeline on data obtained by DNA sequencing of the contents of a mixture of traditional Chinese medicine. The results show the presence of *Dendrobium cruentum*, which is controlled by CITES appendix I under all circumstances, and *Panax ginseng*, for which only the population of the Russian Federation is controlled under appendix II. In the interest of clarity, omitted are the columns for BLAST hit metadata (i.e. query sequence, accession number, % identity, hit length, e-value and bit score)

Query sequence	**Taxon ID**	**Species**	**CITES info**	**Appendix**
OTU_1	55575	*Dioscorea polystachya*		
OTU_2	906701	*Dendrobium cruentum*	*Dendrobium cruentum*	I
OTU_3	540248	*Debregeasia elliptica*		
OTU_4	4054	*Panax ginseng*	*Panax ginseng*, Only the population of the Russian Federation; no other population is included in the Appendices)	II

References

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, *215*, 403–410.

Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J., & Sayers, E. W. (2009). GenBank. *Nucleic acids research*, *37*(Database issue), D26–31. doi:10.1093/nar/gkn723

Coghlan, M. L., Haile, J., Houston, J., Murray, D. C., White, N. E., Moolhuijzen, P., Bellgard, M. I., et al. (2012). Deep Sequencing of Plant and Animal DNA Contained within Traditional Chinese Medicines Reveals Legality Issues and Health Safety Concerns. *PLoS Genetics*, *8*(4). Retrieved from http://www.plosgenetics.org/article/info%3Adoi%2F10.1371%2Fjournal.pgen.1002657

Groenenberg, D. S. J., Neubert, E., & Gittenberger, E. (2011). Reappraisal of the "'Molecular phylogeny of Western Palaearctic Helicidae s.l. (Gastropoda: Stylommatophora)'": When poor science meets GenBank. *Molecular Phylogenetics and Evolution*, *61*, 914–923. Retrieved from http://www.sciencedirect.com/science/article/pii/S1055790311003836

Hebert, P. D. N., Cywinska, A., Ball, S. L., & deWaard, J. R. (2003). Biological identifications through DNA barcodes. *Proceedings of the Royal Society B Biological Sciences*, *270*(1512), 313–321. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/12614582

Pennisi, E. (2008). Proposal to "Wikify" GenBank Meets Stiff Resistance. *Science*, *319*(5870), 1598–9. doi:10.1126/science.319.5870.1598

Savolainen, V., Cowan, R. S., Vogler, A. P., Roderick, G. K., & Lane, R. (2005). Towards writing the encyclopaedia of life : an introduction to DNA barcoding. *Society*, *360*(September), 1805–1811. doi:10.1098/rstb.2005.1730

Shendure, J., & Ji, H. (2008). Next-generation DNA sequencing. *Nature Biotechnology*, *26*(10), 1135–1145. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/18846087

Stoltzfus, A., Lapp, H., Matasci, N., Deus, H., Sidlauskas, B., Zmasek, C. M., Vaidya, G., et al. (n.d.). Phylotastic! Making Tree-of-Life Knowledge Accessible, Re-usable and Convenient. *BMC Bioinformatics*, (In Review).