Your Data: Finding the Forest through the Trees

Article contributed by dtSearch®

You probably have a thicket of different data types: word processing documents, databases, spreadsheets, presentation files, notation files, PDFs, compressed archives, online data formats, emails, etc. To sift through all of that, you could pull up each one in its relevant application—similar to looking tree by tree. Or you could search across everything at once—finding the forest through the trees.

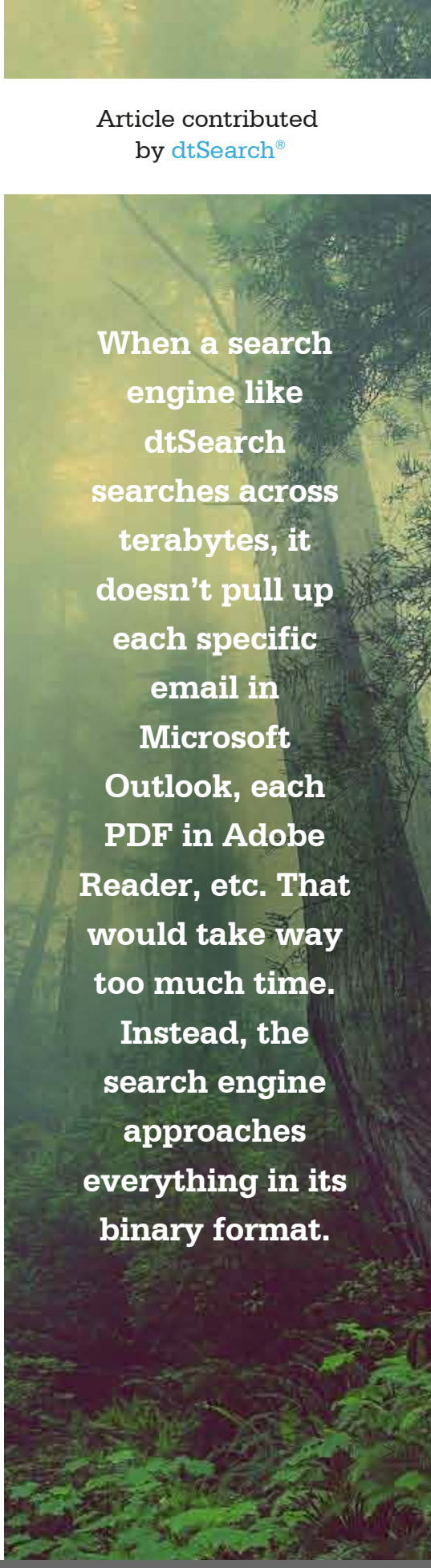
A search engine like dtSearch can give you that full-forest view. dtSearch has enterprise and developer products that run "on premises" or on cloud platforms to instantly search terabytes of "Office" files, PDFs, emails along with nested attachments, databases and online data. Because dtSearch instantly searches terabytes, many dtSearch customers are Fortune 100 companies and government agencies. But anyone with data to search can go to dtSearch.com and download a fully-functional evaluation version.

When a search engine like dtSearch searches across terabytes, it doesn't pull up each specific email in Microsoft Outlook, each PDF in Adobe Reader, etc. That would take way too much time. Instead, the search engine approaches everything in its binary format. That is the version of each file when it is just sitting on a harddrive, network or online server. Each file type has a very different binary encoding.

A search engine needs to figure out which binary encoding to apply for any binary file. Using a filename extension might seem like the obvious choice for determining the correct binary coding to apply. But what if you have a Microsoft Word file saved with a .PDF file extension instead of a .DOCX extension? The more foolproof approach is to look inside each binary file to determine the data type.

The search engine then uses the retrieved data to build an index. An index stores each unique word or number and its precise location in the data. To start indexing with dtSearch, all you have to do is point to as many local drives, network drives, online data repositories, email repositories and the like that you want the search engine to index, and the application will do the rest.

Each dtSearch index can hold up to a terabyte, and there are no limits on the number of indexes that the application can automatically build and then simultaneously search. To handle data updates, you can set the application to automatically update your indexes as often as you want. When dtSearch does that, it checks for anything added, deleted or edited since the last index build. And updating indexes does not in any way block out searching.



Reprinted with permission of USA Daily Times, usadailytimes.com | Original article link (12/17/2020)

While indexing is a very resource-intensive process, searching is not. There are no limits on the number of concurrent search threads running in a network environment, an "on premises" web server or a remote web server such as on Microsoft Azure or AWS.

For precision search, use words or phrases linked with Boolean connectors: "all words" / "and"; "any words" / "or"; as well as an "and not" Boolean option. You can also add proximity searching where one word or phrase appears within X words of another word or phrase in one direction or in either direction. You can make a Boolean and proximity search as precise as you want: forest and (maple trees or pine trees) and not (fruit trees w/12 grow).

The default for ranking retrieved files is natural language vector-space ranking. With that, you can enter a plain English search—or a more structured search—and the application will rank retrieved files by hit term density and rarity. If trees appear in a zillion files, but forest appears in only a handful of files, forest would get a much higher relevancy ranking, letting you literally get right to the forest through the trees.

Fuzzy searching adjusts from 1 to 10 to sift through misspellings, as often appear in OCR'ed PDFs and emails. That way, if forest is mistyped forext, you would still find it with a low level of fuzziness on. Metadata search finds the word forest but only if it appears in specific metadata. Wildcards can hold the place of one or more letters in a word, so pine* would find pineapple. Stemming lets you find different endings on the same route. Phonic searching finds sound-alikes.

Unicode supports works with international language text. dtSearch also has numeric range search and date range search. Advanced users can override the default natural language ranking by assigning positive or negative weighting generally or only if something appears near the top of a file or in specific metadata. Other advanced options include regular expression search and hash value search. dtSearch can even automatically flag credit card numbers mixed in with the text.

Developers using the dtSearch Engine SDK can add faceted search, letting end-users drill down through specific metadata before searching. Filtering lets developers precisely limit who sees what based on metadata in a separate database, metadata in documents, or even the presence or absence of certain keywords.

Please go to dtSearch.com and download a fully-functional 30-day evaluation version. And find the forest through the trees of your own data.

Article contributed by dtSearch®

Filtering lets developers precisely limit who sees what based on metadata in a separate database, metadata in documents, or even the presence or absence of certain keywords.

Reprinted with permission of USA Daily Times, usadailytimes.com | Original article link (12/17/2020)