Internet and Web Applications, Fragenkatalog

Dennis Rössel, Henry Haustein

Was ist TFIDF? Warum wird das heute nicht mehr im Web verwendet?

TFIDF (Term Frequency Inverse Document Frequency) ist eine Kennzahl um die Wichtigkeit eines Begriffes eines Dokumentes in einer Dokumentensammlung zu beschreiben.

- TF: Wie oft taucht der Begriff in dem Dokument auf?
- IDF: In wie vielen Dokumenten der gesamten Dokumentensammlung taucht dieser Begriff auf?

TFIDF beachtet nicht die Struktur der Dokumente und kann recht leicht ausgetrickst werden, indem man Massen an Begriffen unsichtbar in die Dokumente schreibt.

Was macht PageRank anders? Random-Walk-Modell einbeziehen

Bei PageRank geht es um die Summe an Verweisen auf ein Dokument und die Qualität der Seiten, die auf ein Dokument verweisen.

Der Nutzer folgt Hyperlinks (Random-Walk-Modell) und dadurch teilt sich das Prestige einer Seite auf die Seiten, zu denen verwiesen wird, auf. Zudem kann der Nutzer auch mit einer gewissen Wahrscheinlichkeit auf keinen Link mehr klicken. So werden ewige Zyklen verhindert.

Aus welchen Bestandteilen besteht eine Suchmaschine?

Eine Suchmaschine besteht aus:

- Crawler: crawlt Webseiten und speichert sie ins Page repository, folgt Links
- Page repository
- Indexing Module: Indiziert die Webseiten und erstellt einen Index
- Indizes: Content Index und Inverted Index:
  - Content Index: für jedes Dokument werden die wichtigen Wörter gespeichert (tokenization, stemming = Plural entfernen, etc.)
  - Inverted Index: für jeden Term werden die Dokumente, in denen dieser auftaucht, gespeichert
- Ranking Module
- Query Module: Suchfilter

Was ist semantische Suche?

Bei der semantischen Suche untersucht der Algorithmus anhand der Beziehungen der Wörter, Sätze und Texte untereinander, was der Benutzer mit seiner Suche gemeint haben könnte. Daraufhin versucht er, die Antwort auf die gestellte Frage zu finden und sie dem Nutzer direkt anzuzeigen.

Ein Beispiel für eine semantische Suchmaschine ist Wolfram Alpha. Sucht man hier nach *gross domestic product usa* (Bruttoinlandsprodukt der Vereinigten Staaten), liefert die semantische Suche einen Wert von 16,89 Trillionen US-Dollar pro Jahr. Zusätzlich kann der Nutzer jetzt wählen, ob er lieber das nominale oder das quartalsbezogene BIP sehen möchte und so seine Suchanfrage weiter konkretisieren.

Google hingegen resultiert bei derselben Suche 10,7 Mio. Suchergebnisse, doch die Antwort auf seine Frage muss der Nutzer selbst suchen. Versteht die Semantic Search also den Hintergrund einer Fragestellung, so kann sie den Weg zur Erlangung der Information verkürzen.

Quelle: https://de.ryte.com/wiki/Semantic_Search

Was sind CDNs? Wie funktionieren diese technisch?

CDNs sind Content Distributions Networks. Sie halten Kopien von Daten überall auf der Welt vor, sodass diese schnell verfügbar ist. Der passende CDN-Server wird anhand der Nähe zum User und der Auslastung ermittelt.

Der Inhaltsanbieter muss aktiv seinen Inhalt zum CDN schieben, im Gegensatz dazu speichern Proxy Server nur die Daten, die der Nutzer einmal angefordert hat.

Was passiert beim Caching?

Zwischenspeichern von Informationen, die einmal berechnet wurden, sodass diese dem nächsten Nutzer mit der gleichen Anfrage direkt ausgeliefert werden können.

3 Interaktionsmodelle: Erklären + Technologie, Wo ist der Unterschied zwischen den beiden asynchronen Modellen?

Die drei Interaktionsmodelle sind:

- Synchronous communication model: click, wait and refresh
- non blocking communication model: Mittels AJAX können, ohne die Seite neuladen zu müssen, Informationen im Hintergrund abgerufen werden und die Seite kann aktualisiert werden.
- Asynchronous communication model: WebSockets ermöglichen Aktualisierungen im Hintergrund auf schnellerer Basis

Unterschied Comet und WebSockets:

- Bei Comet müssen zwei HTTP-Kanäle aufgebaut werden, dazu sind 2 HTTP-Requests notwendig. Technische Umsetzung über Long Polling (Client sendet Requests, Server antwortet erst, wenn Daten verfügbar sind)
- Bei WebSockets braucht man nur noch einen Verbindung, die mit einem HTTP-Upgrade-Request eingeleitet wird. Dann wird eine TCP-Verbindung aufgebaut.

Bei AJAX muss der Client jedes mal, wenn er ein Update haben will, einen HTTP-Request senden.

Unterschiede zwischen HTTP/1, /1.1, /2 und /3

Verbesserungen von HTTP/1 → HTTP/1.1:

- Persistent connection über `connection: keep-alive`
- Pipelining: nicht warten bis die Antwort eines Requests zurück kommt, sondern schon senden weiterer Requests (Reihenfolge bleibt erhalten)

Verbesserungen von HTTP/1.1 → HTTP/2:

- Multiplexing: ähnlich wie Pipelining, nur Reihenfolge muss nicht erhalten bleiben
- Komprimierung des Headers

Verbesserungen von HTTP/2 → HTTP/3:

- HTTP/3 basiert auf QUIC und nicht mehr auf TCP
- TLS 1.3 auf QUIC-Ebene
- konstante Verbindung → weniger Datenpakete, weil kein Header jedes mal gesendet werden muss
- Fehlerkorrektur auf QUIC-Ebene
- bei Paketverlusten stockt die Verbindung nicht mehr, weil nicht mehr gewartet werden muss
- HTTP/3 verzichtet auf einleitende Handshakes
- HTTP/3 ist nicht mehr an IP-Adressen gebunden, sondern an individuelle Verbindungs-IDs, die selbst bei einem Netzwerkwechsel einen konstanten Download ermöglichen

Unterschiede DOM und SAX

In SAX werden Ereignisse ausgelöst, wenn das XML geparst wird. Wenn der Parser das XML parst und auf ein beginnendes Tag stößt, löst er das Ereignis `tagStarted` (oder ähnliches) aus. Ähnlich verhält es sich, wenn beim Parsen das Ende des Tags erreicht wird, dann wird `tagEnded` ausgelöst. Die Verwendung eines SAX-Parsers setzt voraus, dass man diese Ereignisse verarbeiten und die mit jedem Ereignis zurückgegebenen Daten sinnvoll nutzen kann.

In DOM werden beim Parsen keine Ereignisse ausgelöst. Das gesamte XML wird geparst und ein DOM-Baum (mit den Knoten im XML) wird erzeugt und zurückgegeben. Nach dem Parsen kann der Benutzer durch den Baum navigieren, um auf die verschiedenen Daten zuzugreifen, die zuvor in den verschiedenen Knoten im XML eingebettet waren.

DOM ist im Allgemeinen einfacher zu verwenden, hat aber den Nachteil, dass das gesamte XML geparst werden muss, bevor es verwendet werden kann.

Quelle: https://stackoverflow.com/questions/6828703/what-is-the-difference-between-sax-and-dom

Die 4 wichtigsten Protokolle bei Video- und Audio-Kommunikation: Erklärung

Die Protokolle sind:

- SDP (Session Description Protocol): Port, IP, Codecs
- SIP (Session Initiation Protocol): Gesprächsaufbau, Klingeln, Auflegen, Besetzt, Invite, ... `sip:user@domain`

- RTP (Real-Time Transport Protocol): Gesprächsdaten
- RTCP (Real-Time Control Protocol): QoS, flow control, Fehlerbehebung

Wie wird eine Verbindung mittels SIP aufgebaut? Stichwort: Offer-Answer-Modell, SIP Server

Ein Nutzer schickt ein INVITE und wenn dieses angenommen wird, dann wird RINGING (und später OK wenn angenommen) zurückgesendet. Der Nutzer schickt dann ein ACK mit den SDP-Daten und die RTP-Verbindung wird aufgebaut.

In den meisten Fällen gibt es noch einen Registrar, bei dem sich ein Nutzer anmeldet. Der Registrar kann dann SIP-Namen und IP einer Person liefern, die man anrufen möchte.

Wie können Parameter wie Medientyp, QoS, etc. bei VoIP mitgeteilt/vereinbart werden?

Mittels RTCP werden Statistiken zu Round-Trip-Times und Packetverlusten gesammelt und ermittelt ob Anpassungen an den QoS nötig sind. Wenn ja, werden diese an die RTP-Pakete angehängt.

Generationen des P2P, Was ist Kademlia?, Wie funktioniert BitTorrent?

Generationen des P2P:

- First generation:
  - Centralized P2P: Zentraler Server speichert Liste mit Teilnehmern und welche Dateien diese anbieten
  - Decentralized P2P: Peers sind Server und Client gleichzeitig, zum Beitreten braucht man die IP eines Mitgliedes und findet dann weitere Mitglieder über Broadcast Ping. Um Dateien zu suchen fragt man seine Nachbarn, Query enthält TTL
- Second generation (Hybrid P2P): User fragt nächste Supernode nach Datei (Supernodes sind Server). User kann zur Supernode werden, wenn er z.B. eine besonders gute Verbindung hat
- Third generation (Structured P2P):
  - Kademila (Distributed Hash Tables): Jeder Knoten wird durch eine eindeutige Nummer (genannt "Node-ID") identifiziert. Diese Nummer dient nicht nur zu seiner Identifizierung, sondern wird von Kademlia gleichzeitig für weitere Zwecke herangezogen. Der eigene Knoten berechnet eine zufällige Node-ID, falls er zuvor noch nie im Netz war.

    Ein Knoten, der dem Netz beitreten möchte, muss zuerst einen "Bootstrapping" genannten Prozess durchlaufen: In dieser Phase erhält der Algorithmus von einem Server oder Benutzer im Netzwerk die IP-Adresse einiger Knoten, die bereits im Kademlia-Netz bekannt sind. Von diesen ersten Knoten erhält er nun auch die IP-Adressen weiterer Knoten, so dass keine Abhängigkeit mehr von einzelnen Knoten besteht. Dazu sucht er zunächst nach Knoten, die der eigenen ID ähneln, um sich möglichst günstig dort zu positionieren, wo eine solche ID erwartet wird.

Da es keine zentrale Instanz gibt, die eine Indizierung der vorhandenen Informationen übernimmt, wird diese Aufgabe auf alle Knoten gleichermaßen aufgeteilt. Ein Knoten, der eine Information besitzt, errechnet zuerst den charakteristischen Hashwert ("ID" genannt) dieser Information. Die verwendete Hash-Funktion in einem Kademlia-Netz muss immer dieselbe sein. Jener Knoten sucht nun im Netz die Knoten, deren ID die kleinste "Distanz" zum Hash aufweisen, und übermittelt ihnen seine Kontaktdaten.

Sucht ein Knoten genau diese Information, vollzieht er dieselbe Prozedur und gelangt dadurch an die Knoten, die gespeichert haben, wer im Netz diese Information besitzt. Häufig ist dem Suchenden nur der Hashwert der Daten verfügbar. Er kann nun eine direkte Verbindung zu den Quellen eingehen und die Daten empfangen. Es ist also sichergestellt, dass der Suchende die Kontaktdaten der Quelle genau an der Stelle findet, wo diese sie hinterlassen hat. Da das Netz üblicherweise in ständigem Wandel begriffen ist, werden die Kontaktdaten auf mehrere Knoten verteilt und von der Quelle nach einer gewissen Zeit aktualisiert.

Zum Auffinden eines Knotens hangelt sich der Algorithmus immer näher an diesen heran, bis er gefunden wird oder ein Fehler zurückkommt. Die Anzahl der während dieser Suche maximal zu befragenden Knoten entspricht der Distanz dieses Knotens zu einem selbst. Sollte sich die Anzahl der Teilnehmer im Netz verdoppeln, so muss man nicht doppelt so viele Knoten befragen, sondern pro Verdoppelung nur einen Knoten mehr. Die benötigte Bandbreite skaliert also nicht linear mit der Größe des Netzes, sondern logarithmisch zur Basis zwei.

Ein weiterer Vorteil liegt vor allem in der dezentralen Struktur, die die Resistenz gegen Distributed-Denial-of-Service-Attacken deutlich steigert. Selbst wenn eine ganze Reihe von Knoten angegriffen wird, hat das für das Netz selbst keine allzu großen Auswirkungen. Mit der Zeit strickt sich das Netz dann um diese neuen "Löcher" herum.

Bei Kademlia werden lang bekannte, zuverlässige Knoten beim Routing gegenüber neuen stets bevorzugt und niemals aus den Routing-Tabellen entfernt, weshalb es für einen Angreifer nur schwer möglich ist, die Routing-Struktur des Netzes zu manipulieren.

Die oben genannte "Distanz" hat nichts mit geografischen Gegebenheiten zu tun, sondern bezeichnet die Distanz innerhalb des ID-Bereiches. Es kann also vorkommen, dass z. B. ein Knoten aus Deutschland und einer aus Australien sozusagen "Nachbarn" sind. Die Distanz zwischen zwei Knoten und Hashwerten wird durch die mathematische XOR-Funktion errechnet und beträgt $ID_1 \oplus ID_2$ interpretiert als Integer. (Quelle: https://de.wikipedia.org/wiki/Kademlia)

– BitTorrent: Früher brauchte BitTorrent Tracker, die als Server fungieren (Centralized P2P), heutzutage ist das nicht mehr notwendig, da man mittels DHT auch ohne Tracker die Dateien findet. Allersdings können Tracker die Suche nach Nschbarn beschleunigen, deswegen werden sie immer noch eingesetzt. Die Dateien werden in Chunks aufgeteilt, sodass man verschiedene Chunks einer Datei parallel von mehreren Peers anfordern kann (swarming). Fehlt noch ein Chunk, so wird versucht, diesen von allen Peers herunterzuladen (Endgame Mode)

Was ist der Unterschied zwischen Tagging und Branching bei Versionsverwaltungssystemen?

Tagging: Symbolischer Link zwischen Dateien verschiedener Versionen

Branching: Abspaltung der Codebasis, Weiterentwicklung in eine andere Richtung, eventueller Merge am Ende

Unterschied Git und SVN

Git kann man auf dem lokalen Rechner installieren, während SVN immer eine Server-Client-Architektur braucht.

Binarys sind bei Git problematisch, weil diese immer komplett im Commit sind und damit sich das Repository aufbläst. Bei SVN wird immer nur die aktuelle Version der Binary gespeichert.

weitere Vorteile von Git:

- bei Git braucht man keine Netzwerkverbindung um Git-Operationen durchzuführen, bei SVN schon
- durch lokale Kopien gibt es viele unabhängige Backups des Codes
- Änderungen sind schneller (Merges, Branches, Commits)
- im Open-Source-Bereich sind Änderungen anderer einfacher: Repo forken, Änderungen rein und Pull Request stellen

Was ist XMPP? Was sind Presentities?

XMPP = Extensible Messaging and Presence Protocol (Chatten + Informationen)

Nutzer kann Nachrichten an XMPP-Server senden, dieser leitet diese an Empfänger weiter. Nutzer können auch Informationen an den XMPP-Server senden, wie z.B. Online, Offline, Nicht stören, etc. (Presentities)

Extensible bedeutet, dass man verschiedenen Erweiterungen wie z.B. Videostreams über RTP, Nachrichten über HTTP oder WebSockets anbinden kann.