# Hashing-basierte Indizierung: Anwendungsszenarien, Theorie und Methoden

**Benno Stein** und **Martin Potthast**

Fakultät Medien, Mediensysteme
Bauhaus-Universität Weimar, 99421 Weimar, Germany

benno.stein@medien.uni-weimar.de
martin.potthast@medien.uni-weimar.de

## Abstract

Hashing-basierte Indizierung ist eine mächtige Technologie für die Ähnlichkeitssuche in großen Dokumentkollektionen [Stein 2005]. Sie basiert auf der Idee, Hashkollisionen als Ähnlichkeitsindikator aufzufassen – vorausgesetzt, dass eine entsprechend konstruierte Hashfunktion vorliegt. In diesem Papier wird erörtert, unter welchen Voraussetzungen grundlegende Retrieval-Aufgaben von dieser neuen Technologie profitieren können.

Weiterhin werden zwei aktuelle, hashing-basierte Indizierungsansätze präsentiert und die mit ihnen erzielbaren Verbesserungen bei der Lösung realer Retrieval-Aufgaben verglichen. Eine Analyse dieser Art ist neu; sie zeigt das enorme Potenzial maßgeschneiderter hashing-basierter Indizierungsmethoden wie zum Beispiel dem Fuzzy-Fingerprinting.

## 1 Hintergrund

Vereinfachend gesagt behandelt das textbasierte Information-Retrieval die zielgerichtete Suche in einer große Menge $D$ von Dokumenten. In diesem Zusammenhang unterscheiden wir zwischen dem „realen“ Dokument $d \in D$ in Form eines Papiers, eines Buchs oder einer Web-Seite, und seiner (Computer)repräsentation $\mathbf{d}$ in Form eines Wortvektors, eines Suffixbaums oder einer Signaturdatei. Sei $\mathbf{D}$ die Menge der Repräsentationen aller realen Dokumente aus $D$.

In vielen Anwendungen ist die (Computer)repräsentation $\mathbf{d}$ eines Dokuments ein $m$-dimensionaler Vektor, so dass sich die Objekte in $\mathbf{D}$ als Elemente eines $m$-dimensionalen Vektorraums auffassen lassen. Die Ähnlichkeit zwischen zwei Dokumenten $d_1, d_2$ sei als umgekehrt proportional zur Distanz zwischen den Vektoren $\mathbf{d_1}, \mathbf{d_2} \in \mathbf{D}$ vereinbart. Zur Messung der Ähnlichkeit dient eine Funktion $\varphi(\mathbf{d_1}, \mathbf{d_2})$, die auf das Intervall $[0; 1]$ abbildet, wobei 0 keine Ähnlichkeit und 1 maximale Ähnlichkeit bedeutet. $\varphi$ kann zum Beispiel auf der $l_1$-Norm, der $l_2$-Norm oder auf dem Winkel zwischen zwei Vektoren beruhen.

Offensichtlich maximiert das zu $d$ ähnlichste Dokument $d' \in D$ die Funktion $\varphi(\mathbf{d}, \mathbf{d}')$, und offensichtlich kann $d'$ durch eine lineare Suche in $\mathbf{D}$ gefunden werden. Weniger bekannt ist, dass sich die Bestimmung von $d'$ nicht schneller als in $O(|\mathbf{D}|)$ bewerkstelligen lässt, falls die Dimensionalität $m$ des Vektorraums etwa 10 oder mehr beträgt [Weber *et al.* 1998]. An diesem Punkt setzt die Idee der hashing-basierten Indizierung an: Mithilfe von Hashing lässt sich in quasi konstanter Zeit feststellen, ob $\mathbf{d}$ ein Element in $\mathbf{D}$ ist. Dieses Konzept ist auf die Ähnlichkeitssuche übertragbar, falls eine Hashfunktionen $h_\varphi : \mathbf{D} \rightarrow U$ existiert, die eine Menge $\mathbf{D}$ von Dokumentrepräsentationen auf ein Universum $U, U \subset \mathbf{N}$ von Hashwerten abbildet und die folgende Eigenschaft besitzt [Stein 2005]:

$$h_\varphi(\mathbf{d}) = h_\varphi(\mathbf{d}') \;\Rightarrow\; \varphi(\mathbf{d}, \mathbf{d}') \geq 1 - \varepsilon, \qquad (1)$$

mit $\mathbf{d}, \mathbf{d}' \in \mathbf{D}$ und $0 < \varepsilon \ll 1$. Anders ausgedrückt, eine Hashkollision zwischen zwei Elementen aus $\mathbf{D}$ kann als Indiz für eine hohe Ähnlichkeit zwischen ihnen gewertet werden.

Aufbauend auf dieser Idee werden in Abschnitt 2 Anwendungsszenarien diskutiert, bei denen sich durch hashing-basierte Indizierung die Retrieval-Performanz und die Ergebnisqualität signifikant verbessern lässt. Abschnitt 3 stellt zwei hashing-basierte Indizierungsverfahren vor, die im textbasierten Information-Retrieval anwendbar sind, und Abschnitt 4 präsentiert Ergebnisse einer vergleichenden Analyse der beiden Ansätze für ausgewählte Retrieval-Aufgaben.

## 2 Aufgaben und Anwendungsszenarien im textbasierten Information-Retrieval

Sei $T$ die Menge aller Worte, die in den Dokumentrepräsentationen $\mathbf{d} \in \mathbf{D}$ verwendet werden. Die wichtigste Datenstruktur zur Indizierung von $D$ ist die invertierte Liste [Witten *et al.* 1999; Baeza-Yates und Ribeiro-Neto 1999]. Jedes Wort $t \in T$ wird auf eine sogenannte Vorkommensliste abgebildet, die für jedes Vorkommen von $t$ in den jeweiligen Dokumenten $d_i$, $i = 1, \ldots, o$, eine eindeutige Referenz auf die entsprechende Position innerhalb von $d_i$ speichert.

Sei eine – möglicherweise sehr große – Dokumentkollektion $D$ gegeben, die sowohl durch eine invertierte Liste, $\mu_i$, als auch durch einen Hashindex, $\mu_h$, indiziert ist:

$$\mu_i : T \rightarrow \mathcal{D}$$
$$\mu_h : \mathbf{D} \rightarrow \mathcal{D}$$

$\mathcal{D}$ bezeichnet die Potenzmenge von $D$. Während die invertierte Liste $\mu_i$ verwendet wird, um Wortanfragen zu beantworten, ist der Hashindex $\mu_h$ besonders dazu geeignet, Anfragen zu behandeln, die in Form eines Beispieldokuments formuliert sind. Bei dieser Art von Anfragen werden alle zu dem Beispieldokument ähnlichen Dokumente gesucht. Abschnitt 3 zeigt, wie ein entsprechender Hashindex zusammen mit einer passenden Hashfunktion $h_\varphi$ konstruiert werden kann.

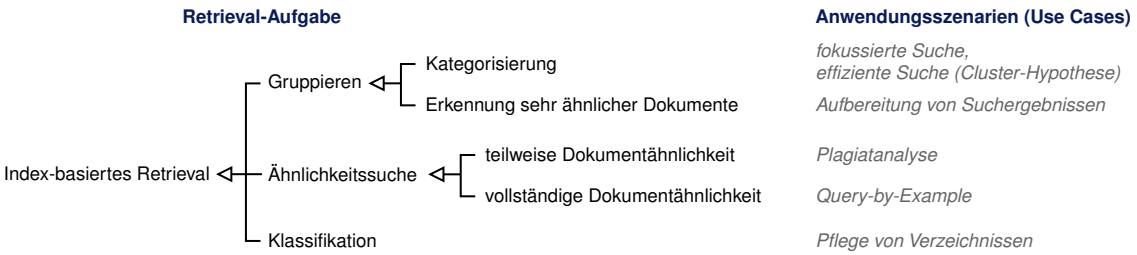


Abbildung 1: Taxonomie von Aufgaben und Beispiele für Anwendungsszenarien im textbasierten Information-Retrieval, die sich durch hashing-basierte Indizierung verbessern lassen.

Wir haben drei Arten von Retrieval-Aufgaben identifiziert, bei denen hashing-basierte Indizierung Verbesserungspotenzial birgt: (*i*) Gruppierung, insbesondere die Dokumentkategorisierung und die Identifikation sehr ähnlicher Dokumente (near duplicate identification [Broder 2000]), (*i*) Ähnlichkeitssuche, wobei zwischen vollständigen und partiellen Vergleichsmethoden unterschieden werden sollte, und (*iii*) Klassifikation. Abbildung 1 organisiert diese Aufgaben und nennt Beispiele für Anwendungsszenarien; die folgenden Unterabschnitte erläutern das Potenzial der hashing-basierten Indizierung aus technischer Sicht.

## 2.1 Retrieval-Aufgabe: Gruppierung

Die Gruppierung von Dokumenten spielt eine wichtige Rolle bei der Benutzerinteraktion und bei Schnittstellen von Informationssystemen: Viele Retrieval-Aufgaben liefern eine große Ergebnismenge mit Dokumenten, die sortiert, visuell aufbereitet oder von Duplikaten befreit werden soll. Eine Sortierung oder eine visuelle Aufbereitung erfordern die Identifikation von geeigneten Kategorien, also die Lösung eines unüberwachten Klassifikationsproblems. Kategorisierende Suchmaschinen wie Vivísimo und AIsearch lösen diese Aufgabe mittels einer Cluster-Analyse [Zamir und Etzioni 1998; Meyer zu Eißen und Stein 2002]. Die Erkennung von annähernd identischen Dokumenten ist nützlich u. a. bei der Produktsuche im World Wide Web oder zur Eliminierung von Dokumenten, die – bedingt durch die Spiegelung von Web-Seiten – mehrfach in einer Ergebnismenge auftauchen.

*Anmerkungen zur Laufzeit.* Durch hashing-basierte Indizierung lässt sich die Laufzeit solcher Retrieval-Aufgaben deutlich verkleinern. Ein Informationsbedarf wird hier als Wortanfrage formuliert, für die zunächst mit einer invertierten Liste $\mu_i$ die Ergebnismenge $D' \subseteq D$ ermittelt wird. Anschließend kommt ein Hashindex $\mu_h$ zur Gruppierung zum Einsatz. Abbildung 2 illustriert die Strategie. Der Hashindex $\mu_h$ erlaubt die Gruppierung der Dokumente in Linearzeit in der Größe der Ergebnismenge $|D'|$. Die Verwendung eines vektorbasierten Dokumentmodells hätte eine Laufzeit von $O(|D'|^2)$ zur Folge, da Duplikaterkennung oder Kategorisierung eine paarweise Ähnlichkeitsberechnung zwischen allen Elementen in $D'$ erfordert.

## 2.2 Retrieval-Aufgabe: Ähnlichkeitssuche

Die am weitesten verbreitete Retrieval-Aufgabe ist diejenige Ähnlichkeitssuche, bei der Anwender ihren Informationsbedarf als Wortanfrage formulieren. Falls alle zur Wortanfrage passenden Dokumente zu ermitteln sind, ist eine invertierte Liste $\mu_i$ der optimale Index für die zu durchsuchende Kollektion. Bekannte Suchmaschinen wie Google, Yahoo oder AltaVista sind darauf spezialisiert, diese Art

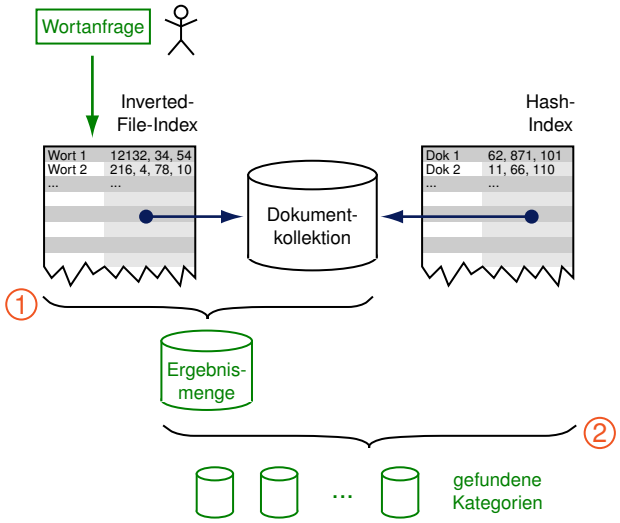


Abbildung 2: Illustration der Retrieval-Aufgabe Gruppierung. Ausgangspunkt ist ein als Wortanfrage formulierter Informationsbedarf. Mit einer invertierten Liste wird die Ergebnismenge derjenigen Dokumente ermittelt, die zu der Wortanfrage passen (Schritt ①). Anschließend wird mit einem Hashindex die Ergebnismenge in Linearzeit kategorisiert (Schritt ②).

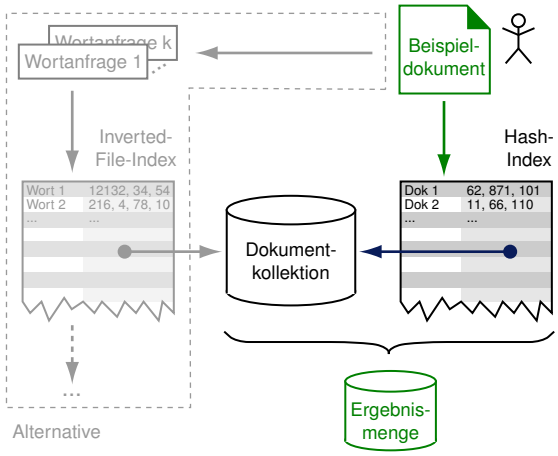


Abbildung 3: Illustration der Retrieval-Aufgabe Ähnlichkeitssuche. Ausgangspunkt ist ein in Form eines Beispieldokuments formulierter Informationsbedarf, der sich mit einem Hashindexes in konstanter Zeit beantworten lässt. Ein alternativer Ansatz (links angedeutet) erfordert die Extraktion von Schlüsselworten aus dem Beispieldokument sowie die Konstruktion geeigneter Wortanfragen.

von Anfragen extrem effizient zu bedienen.

Beschreibt ein Anwender seinen Informationsbedarf mit einem Beispieldokument („Suche ähnliche Dokumente“), kann ein hashing-basierter Index $\mu_h$ zum Einsatz kommen. Voraussetzung hierfür ist, dass eine Hashfunktion mit der Eigenschaft (1) konstruiert werden kann. Dies wiederum hängt von dem akzeptierten $\varepsilon$-Intervall des Anwenders für seinen speziellen Informationsbedarf ab.

*Anmerkungen zur Laufzeit.* Verglichen mit einer invertierten Liste $\mu_i$ ist mit einem hashing-basierten Index $\mu_h$ die Retrieval-Aufgabe um Größenordnungen schneller lösbar. Um die zu einem Beispieldokument ähnlichen Dokumente unter Verwendung von $\mu_i$ zu finden, wären zunächst geeignete Schlüsselworte aus dem Beispieldokument zu extrahieren, eine Reihe von $k$ Wortanfragen zu formulieren und die Ergebnismengen $D'_1, \ldots, D'_k$ mit dem Beispieldokument zu vergleichen. Abbildung 3 (links angedeutet) illustriert diese Strategie; auf der rechten Seite ist die Strategie unter Verwendung von $\mu_h$ gezeigt. Unter der Annahme, dass die Zugriffszeit für beide Indizierungskonzepte $O(1)$ beträgt, benötigt die Konstruktion der Ergebnismenge bei Verwendung von $\mu_i$ eine Laufzeit von $O(|D'_1|+\ldots+|D'_k|)$, bei Verwendung von $\mu_h$ aber nur eine Laufzeit von $O(1)$.

Der tatsächliche erzielte Laufzeitunterschied hängt von der Qualität der extrahierten Schlüsselworte ab, sowie dem Geschick, hieraus Wortanfragen zu formulieren. Die Praxis zeigt, dass beträchtliche Verbesserungen zu erwarten sind. Dieses Ergebnis wird bei Retrieval-Aufgaben wie der Plagiatanalyse weiterhin verbessert: Hier ist die Segmentierung des Eingabedokuments – und damit eine Vervielfachung der Anfragen – notwendig, da eine Ähnlichkeitssuche für jeden einzelnen Abschnitt zu erfolgen hat [Stein und Meyer zu Eißen 2006].

### 2.3 Retrieval-Aufgabe: Klassifikation

Klassifikation spielt eine wichtige Rolle bei vielen textbasierten Retrieval-Aufgaben wie der Genre-Analyse, dem Filtern von Spam-Mails, der Kategoriezuordnung oder der Authoridentifikation. Es handelt sich hierbei um überwachte Klassifikationsaufgaben, also dem Pendant zur unüberwachten Kategorie*bildung*; sie lässt sich – eine kleine Anzahl von Klassen vorausgesetzt – erfolgreich mit Bayes, Diskriminanzanalyse, Support-Vector-Machines oder Neuronalen Netzen lösen. Bei einer großen Anzahl Klassen ist die Konstruktion eines Klassifikators mit garantierten statistischen Eigenschaften nahezu unmöglich.

Mit hashing-basierter Indizierung kann für eine Menge von Klassen $C_1, \ldots, C_p$ ein robuster Klassifikator konstruiert werden, selbst wenn $p$ groß ist oder wenn nur eine kleine oder unregelmäßig verteilte Menge von Trainingsdokumenten vorliegt. Der Klassifikationsansatz folgt dem Prinzip der Nächsten-Nachbar-Suche und geht davon aus, dass die Trainingsdokumente der Klassen $C_i$ in einem Hashindex $\mu_h$ indiziert sind. Für ein neu zu klassifizierendes Dokument $d'$ wird der Hashwert berechnet und die Menge $D'$ derjenigen Dokumente ermittelt, die demselben Hash-Bucket wie $d'$ zugeordnet sind. Diese Dokumente werden bezüglich ihrer Verteilung in $C_1, \ldots, C_p$ untersucht und $d'$ wird derjenigen Klasse $C_j$ zugeordnet, der die meisten der Dokumente aus $D'$ angehören:

$$C_j = \underset{i=1,\ldots,p}{\operatorname{argmax}} |C_i \cap D'|$$

## 3 Hashing-basierte Indizierungsverfahren

Ein hashing-basierter Index $\mu_h$ kann auf Basis einer Hashfunktion $h_\varphi : \mathbf{D} \rightarrow U$ direkt mit einer Hashtabelle $\mathcal{T}$ und einer Standardhashfunktion $h : U \rightarrow \{1, \ldots, |\mathcal{T}|\}$ konstruiert werden. Dabei bildet $h$ das Universum $U$ von Hashwerten gleichmäßig auf die $|\mathcal{T}|$ Speicherstellen der Hashtabelle ab.

Um eine Menge $\mathbf{D}$ von Dokumentrepräsentationen zu indizieren, wird der Hashwert $h_\varphi(\mathbf{d})$ aller Dokumente $\mathbf{d} \in \mathbf{D}$ berechnet und in $\mathcal{T}$ an Speicherstelle $h(h_\varphi(\mathbf{d}))$ eine Referenz auf $d$ gespeichert. $\mathcal{T}$ enthält also für jeden Hashwert von $h_\varphi$ einen Hash-Bucket $D' \subset D$ mit der folgenden Eigenschaft:

$$d_1, d_2 \in D' \;\Rightarrow\; h_\varphi(\mathbf{d_1}) = h_\varphi(\mathbf{d_2}),$$

wobei $\mathbf{d_1}, \mathbf{d_2}$ die Dokumentrepräsentationen der Dokumente $d_1, d_2$ bezeichnen. Mit $\mathcal{T}$ und $h_\varphi$ kann eine als Beispieldokument formulierte Anfrage durch einmaliges Nachschlagen in der Hashtabelle in $O(1)$ beantwortet werden. Erfüllt $h_\varphi$ weiterhin die Eigenschaft (1), so entspricht der für $\mathbf{d}$ ermittelte Hash-Bucket der Menge $\mathbf{D}'$ von Dokumenten, die sich bezüglich $\varphi$ in der $\varepsilon$-Umgebung von $\mathbf{d}$ befinden:

$$\mathbf{d}' \in \mathbf{D}' \;\Rightarrow\; \varphi(\mathbf{d}, \mathbf{d}') \geq 1 - \varepsilon$$

Die wesentliche Herausforderung besteht in der Wahl und der Parametrisierung einer geeigneten Ähnlichkeitshashfunktion $h_\varphi$ für Textdokumente. Zwei kürzlich vorgeschlagene Ansätze sind für diese Aufgabe geeignet: Fuzzy-Fingerprinting und Locality-Sensitive-Hashing.

### 3.1 Fuzzy-Fingerprinting

Bei Fuzzy-Fingerprinting handelt es sich um einen Hashing-Ansatz, der speziell für das textbasierte Information-Retrieval entwickelt wurde – jedoch nicht darauf beschränkt ist [Stein 2005]. Es basiert auf der Definition einer kleinen Anzahl $k$, $k \in [10, 40]$, von Präfixäquivalenzklassen. Eine Präfixäquivalenzklasse enthält alle Worte, die mit demselben Präfix beginnen. Die Berechnung von $h_\varphi(\mathbf{d})$ geschieht in den folgenden Schritten: (*i*) Bestimmung von $\mathbf{pf}$, einem $k$-dimensionalen Vektor, der die Verteilung der Indexterme in $\mathbf{d}$ auf die $k$ Präfixäquivalenzklassen quantifiziert. (*ii*) Normalisierung von $\mathbf{pf}$ auf Basis eines repräsentativen Korpus und Berechnung von $\Delta_{\mathbf{pf}} = (\delta_1, \ldots, \delta_k)^T$, dem Vektor der Abweichungen von der erwarteten Verteilung.[1] (*iii*) Fuzzifizierung von $\Delta_{\mathbf{pf}}$ durch die Projektion der exakten Abweichungen unter Verwendung verschiedener Fuzzifizierungsschemata. Abbildung 4 illustriert die Berechnungsvorschrift.

In der Praxis kommen zwei bis drei Fuzzifizierungsschemata zum Einsatz, wobei jedes Schema (= linguistische Variable) bis zu vier Intervalle umfasst. Gleichung 2 definiert, wie sich aus dem normalisierten Abweichungsvektor $\Delta_{\mathbf{pf}}$ eines Dokuments unter Verwendung eines Fuzzifizierungsschemas $\rho$, das $r$ Intervalle besitzt, ein Hashwert berechnet:

$$h_\varphi^{(\rho)}(\mathbf{d}) = \sum_{i=0}^{k-1} \delta_i^{(\rho)} \cdot r^i, \quad \text{mit } \delta_i^{(\rho)} \in \{0, \ldots, r-1\} \quad (2)$$

$\delta_i^{(\rho)}$ ist ein dokumentspezifischer Wert und beschreibt die fuzzifizierte Abweichung von $\delta_i \in \Delta_{\mathbf{pf}}$ vom Erwartungswert unter Verwendung des Fuzzifizierungsschemas $\rho$.

[1] Als Referenzkorpus dient uns der British National Corpus. Er enthält ca. 100 Millionen Worte und repräsentiert einen aktuellen Querschnitt der geschriebenen und gesprochenen englischen Sprache [Aston und Burnard 1998].

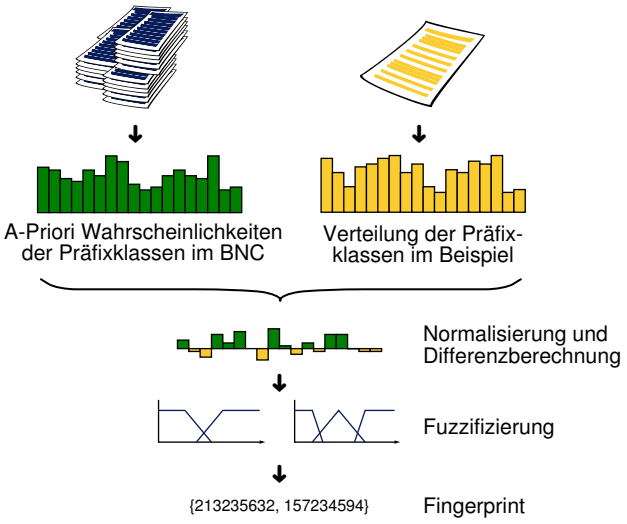


Abbildung 4: Die Berechnung eines Hashwerts durch Fuzzy-Fingerprinting.

Dokumentrepräsentationen auf Basis von Präfixäquivalenzklassen lassen sich als Abstraktionen des Vektorraummodells auffassen. Einerseits schneidet ein auf diese Art abstrahiertes Dokumentmodell bei den Retrieval-Aufgaben Gruppierung, Ähnlichkeitssuche oder Klassifikation tendenziell schlechter ab als das Vektorraummodell; andererseits sind die entsprechenden Vektoren um Größenordnungen kleiner und nicht dünn besetzt.

### 3.2 Locality-Sensitive-Hashing

Locality-Sensitive-Hashing (LSH) stellt einen allgemeinen Rahmen für die Konstruktion von Hashfunktionen dar [Indyk und Motwani 1998]. Eine lokalitätssensitive Hashfunktion $h_\varphi$ ist eine Kombination von $k$ einfachen Hashfunktionen $h$, $h : \mathbf{D} \rightarrow U$, die zufällig und unabhängig voneinander aus einer Familie $H_\varphi$ von Hashfunktionen gezogen sind. Wählt man die Addition als Verknüpfungsoperator, so berechnet sich der Hashwert $h_\varphi(\mathbf{d})$ durch Addition der Hashwerte der einfachen Hashfunktionen:

$$h_\varphi(\mathbf{d}) = \sum_{i=1}^{k} h_i(\mathbf{d}), \qquad \text{mit } \{h_1, \ldots, h_k\} \subset_{rand} H_\varphi$$

In der letzten Zeit sind verschiedene Familien $H_\varphi$ von Hashfunktionen entwickelt worden, die sich im textbasierten Information-Retrieval anwenden lassen [Charikar 2002; Datar *et al.* 2004; Bawa *et al.* 2005]; hier konzentrieren wir uns auf den Ansatz von Datar *et al.* Die Idee dieser Hashfamilie ist, eine Dokumentrepräsentation $\mathbf{d}$ durch Berechnung des Skalarprodukts $\mathbf{a}^T \cdot \mathbf{d}$ auf eine reelle Zahl abzubilden. $\mathbf{a}$ ist ein Zufallsvektor, dessen Vektorkomponenten unabhängig voneinander aus einer bestimmten Wahrscheinlichkeitsverteilung gezogen sind. Der reelle Zahlenstrahl wird in äquidistante Intervalle der Breite $r$ unterteilt und jedem Intervall eine eindeutige natürliche Zahl zugewiesen. Das Skalarprodukt wird mit der Zahl desjenigen Intervalls assoziiert, in das der berechnete Wert des Skalarprodukts fällt. Die Berechnung von $h_\varphi$ für $k$ Zufallsvektoren $\mathbf{a}_1, \ldots, \mathbf{a}_k$ geschieht wie folgt:

$$h_\varphi^{(\rho)}(\mathbf{d}) = \sum_{i=1}^{k} \left\lfloor \frac{\mathbf{a}_i^T \cdot \mathbf{d} + c}{r} \right\rfloor$$

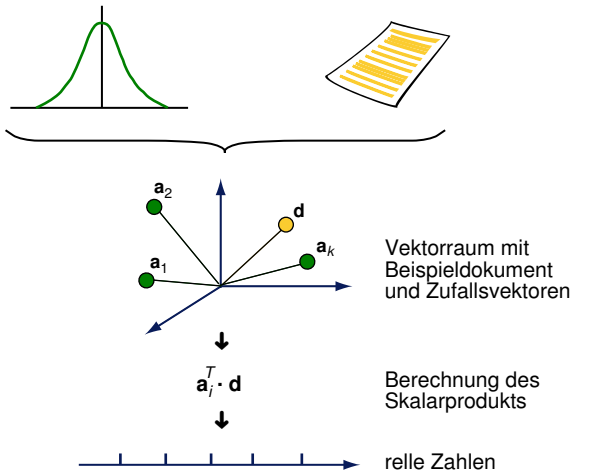


Abbildung 5: Die Berechnung eines Hashwerts durch Locality-Sensitive-Hashing.

$c \in [0, r]$ wird zufällig gewählt, um alle Segmentierungen des reellen Zahlenstrahls zu ermöglichen. Abbildung 5 illustriert die Berechnungsvorschrift.

Eine besondere Eigenschaft von Locality-Sensitive-Hashing ist, dass eine untere Schranke für die Retrieval-Qualität angegeben werden kann: Ist die durchschnittliche Distanz eines Dokuments zu seinem nächsten Nachbarn a-Priori bekannt, lässt sich $h_\varphi$ so parametrisieren, dass die Wahrscheinlichkeit, den nächsten Nachbarn zu finden, oberhalb eines bestimmten Grenzwerts liegt [Gionis *et al.* 1999]. Diese Eigenschaft folgt aus der lokalen Sensitivität der zugrundeliegenden Hashfamilie $H_\varphi$ und schreibt vor, dass für alle $h \in H_\varphi$ die Wahrscheinlichkeit einer Kollision der Hashwerte zweier Dokumente mit deren Ähnlichkeit steigt.[2]

### 3.3 Retrieval-Eigenschaften von Ähnlichkeitshashfunktionen

Auffälligstes Merkmal hashing-basierter Indizierungsverfahren ist die Abstraktion eines feingranularen Ähnlichkeitskonzeptes – quantifiziert durch eine Ähnlichkeitsfunktion $\varphi$ – auf das binäre Konzept „ähnlich oder nicht ähnlich“: Zwei Dokumentrepräsentationen werden als ähnlich angesehen, wenn ihre Hashwerte gleich sind; andernfalls wird angenommen, dass sie nicht ähnlich sind. Diese als Eigenschaft (1) formalisierte Implikation steht in direkter Beziehung zu dem statistischen Konzept der *Precision*. Die Umkehrung der Implikation steht in direkter Beziehung zu dem statistischen Konzept des *Recall*: Ist die Ähnlichkeit zweier Dokumentrepräsentationen größer als ein bestimmter Schwellwert $1 - \varepsilon$, so wird angenommen, dass ihre Hashwerte gleich sind.

Es ist zu bemerken, dass Letzteres für eine Hashfunktion $h_\varphi$ nicht in der Allgemeinheit gelten kann. $h_\varphi$ berechnet für jede Dokumentrepräsentation genau einen Hashwert und definiert dadurch eine absolute Partitionierung des Raums der Dokumentrepräsentationen.[3] Zwangsläufig muss der

[2] Die Hashfamilie von Datar *et al.* ist lokalitätssensitiv, wenn die eingesetzte Wahrscheinlichkeitsverteilung $\alpha$-stabil ist; das bekannteste Beispiel einer solchen Verteilung ist die Gauss-Verteilung. Grundlagen hierzu und weitere Details sind in Indyk [2000] und Nolan [2005] beschrieben.

[3] Im Gegensatz dazu definiert der vollständige Ähnlichkeitsgraph einer Menge $\mathbf{D}$ von Dokumentrepräsentationen für jedes Element in $\mathbf{D}$ eine spezifische Partitionierung.

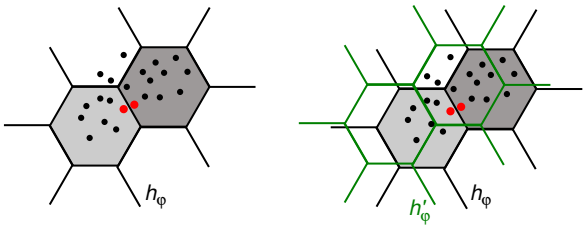


Abbildung 6: Abbildung einer Menge von Dokumentrepräsentationen in die Ebene. Eine Hashfunktion $h_\varphi$ unterteilt die Ebene in Regionen, wobei jede Region durch genau einem Hashwert charakterisiert ist. Auch zwei sehr ähnliche Dokumentrepräsentationen (rot markiert) können auf verschiedene Hashwerte abgebildet sein (links dargestellt). Diese Schwellwertcharakteristik lässt sich durch die Verwendung mehrerer Hashfunktionen $h_\varphi$ und $h'_\varphi$ abmildern (rechts dargestellt).

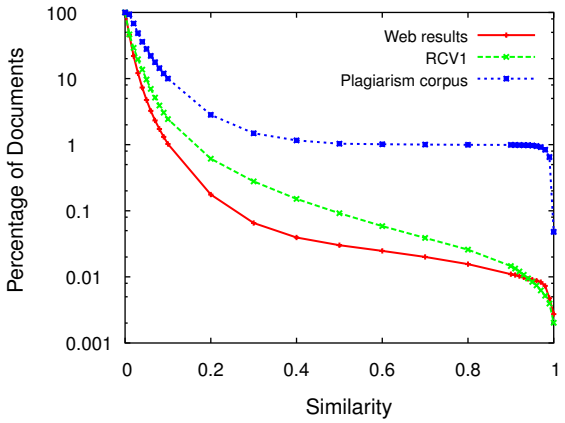


Abbildung 7: Das Diagramm zeigt das Verhältnis der Dokumente, deren paarweise Ähnlichkeit über einem bestimmten Schwellwert liegt.

durchschnittliche Recall zu einer Suchanfrage kleiner als 1 sein. Abbildung 6 illustriert diesen Zusammenhang: Trotz ihrer hohen Ähnlichkeit (= geringe Distanz) bildet die Hashfunktion $h_\varphi$ einige der Dokumentrepräsentationen auf verschiedene Hashwerte ab. Wird zusätzlich eine zweite Hashfunktion $h'_\varphi$ verwendet, die den Raum leicht unterschiedlich partitioniert, kann eine Suchanfrage durch die disjunktive Verknüpfung der beiden Hashfunktionen beantwortet werden. In der Praxis entspricht das der Konstruktion von zwei Hashindizes $\mu_h, \mu'_h$ und der Rückgabe der Vereinigungsmenge beider Ergebnismengen als Gesamtergebnismenge einer Suchanfrage. Tatsächlich lässt sich ein monotoner Zusammenhang zwischen der Anzahl der Hashfunktionen und dem erzielten Recall beobachten. Ein so verbesserter Recall geht auf Kosten der Precision.

Es ist aufschlussreich, die verschiedenen Konzepte zu vergleichen, mit denen Varianz in die Hashwertberechnung eingebracht wird: Fuzzy-Fingerprinting verwendet hierfür unterschiedliche Fuzzifizierungsschemata $\rho_i$, Locality-Sensitive-Hashing verwendet hierfür unterschiedliche Mengen von Zufallsvektoren $\rho_i$. In beiden Fällen wird eine Dokumentrepräsentation $\mathbf{d}$ durch eine *Menge* von $l$ einzelnen Hashwerten $\{h_\varphi^{(\rho_i)}(\mathbf{d}) \mid i = 1, \ldots, l\}$ kodiert. Diese Menge bezeichnen wir als Fingerabdruck.

# 4 Fuzzy-Fingerprinting versus LSH: Fallstudien

Dieses Kapitel präsentiert einige Ergebnisse umfassender Experimente, in denen die beiden Ansätze zur hashingbasierten Indizierung für die Retrieval-Aufgaben der Duplikateliminierung bzw. der Identifikation fast gleicher Dokumente (Abschnitt 2.1) und der Ähnlichkeitssuche (Abschnitt 2.2) eingesetzt wurden. Die Experimente zeigen die Alltagstauglichkeit dieser Technologie und erlauben Aussagen dahingehend, welcher der Ansätze besser für die jeweilige Retrieval-Aufgabe bzw. für das textbasierte Information-Retrieval im Allgemeinen geeignet ist.

## 4.1 Aufbau der Experimente

Die Experimente wurden auf Grundlage von drei Testkollektionen durchgeführt; zwei der Kollektionen enthalten jeweils 100.000 Dokumente, die dritte enthält 3.000 Dokumente.[4]

[4]Die Metadateien zur Beschreibung der Kollektionen stellen wir anderen Wissenschaftlern auf Anfrage zur Verfügung.

Die erste Kollektion (Web) wurde mit den Suchmaschinen Yahoo, Google und AltaVista erstellt und enthält die Ergebnisse einer fokussierten Suche. Hierzu wurde zunächst eine kleine Menge von Dokumenten über ein bestimmtes Thema ausgewählt und hieraus etwa 100 Schlüsselworte mittels einer Kookkurenzanalyse extrahiert (vgl. Matsuo und Ishizuka [2004]). Diese Vorgehensweise soll eine unverzerrte Auswahl von Schlüsselworten für ein Thema sicherstellen. Auf Basis der Schlüsselwortmenge wurden aus bis zu fünf Worten bestehende Anfragen generiert und den Suchmaschinen übergeben. Für jede Anfrage wurden die am höchsten eingestuften Suchergebnisse geladen und der Textinhalt extrahiert. Diese Kollektion dient zur Nachbildung von Ergebnismengen, wie sie von typischen Web-Retrieval-Systemen geliefert werden.

Die zweite Kollektion ist eine Auswahl von Dokumenten aus dem „Reuters Corpus Volume 1“ (RCV1), der von der Reuters Corporation für Forschungszwecke veröffentlicht wurde [Rose *et al.* 2002]. Der Korpus enthält mehr als 800.000 Dokumente, von denen jedes zwischen einigen hundert bis zu mehreren tausend Worten umfasst. Die Dokumente sind mit Metainformationen wie Kategorie, geographische Region oder Industriesektor angereichert. Insgesamt gibt es 103 verschiedene Kategorien, die hierarchisch unter den vier Hauptkategorien „Government, Social“, „Economics“, „Markets“ und „Corporate, Industrial“ einsortiert sind. Jede der Hauptkategorien ist die Wurzel eines Baums von Unterkategorien, so dass jede Unterkategorie die Informationen seiner Elternkategorie verfeinert. Diese Kollektion dient zur Nachbildung von Retrieval-Situationen in Unternehmen, die ihre Dokumente in vordefinierten Verzeichnishierarchien organisieren.

Die dritte Kollektion ist ein speziell angefertigter Korpus, um verschiedene Arten von Plagiatvergehen zu simulieren.[5] Die darin enthaltenen Dokumente wurden mit einem Algorithmus zur Synthese von Plagiatinstanzen erzeugt; Dokumente der ACM Digital Library bildeten die Eingabe. Diese Kollektion dient zur Nachbildung von Retrieval-Situationen, in denen es gilt, sehr ähnliche Dokumente zu entdecken.

[5]„Ein Plagiat ist die Vorlage fremden geistigen Eigentums bzw. eines fremden Werkes als eigenes Werk oder Teil eines eigenen Werkes“ Wikipedia [2006].

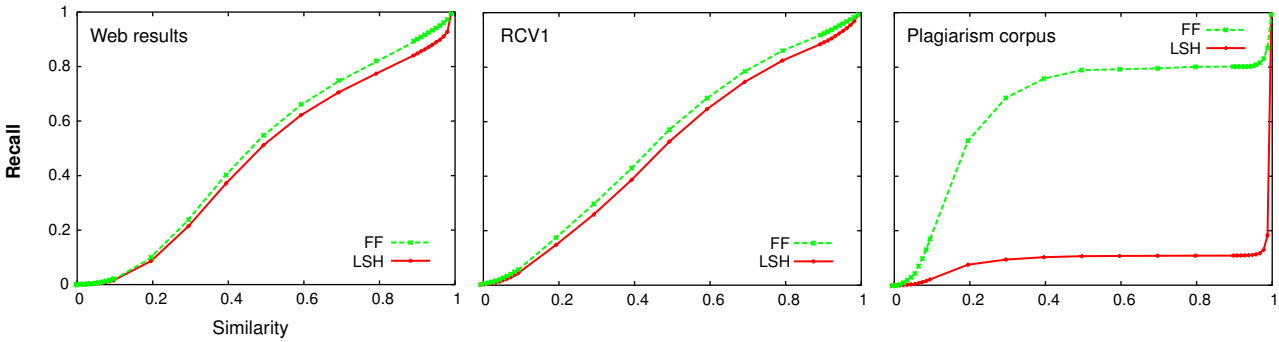


Abbildung 8: Der mit Fuzzy-Fingerprinting (FF) und Locality-Sensitive-Hashing (LSH) auf den drei Testkollektionen erzielte Recall in Abhängigkeit von der Ähnlichkeit.

## 4.2 Ergebnisse

Um die Retrieval-Performanz der Ähnlichkeitshashfunktionen zu messen, wurden für jede Testkollektion die Hashindizes gemäß Fuzzy-Fingerprinting und Locality-Sensitive-Hashing konstruiert. Für jeden Hashindex wurden für die Ähnlichkeitsschwellwerte $0.1 \cdot i,\ i \in \{0, \ldots, 10\}$, die durchschnittliche Precision und der durchschnittliche Recall ermittelt. Hierzu wurden Anfragen ausgewertet, bei denen jedes Dokument einer Kollektion als Beispieldokument diente. Die Referenzwerte für Precision und Recall basierten auf dem Vektorraummodell, dem $tf \cdot idf$-Schema und der Anwendung des Kosinusähnlichkeitsmaßes.

Die Analyse der drei Testkollektionen zeigt, dass der Anteil der Dokumente einer Kollektion, deren paarweise Ähnlichkeit über einer bestimmten Ähnlichkeitsschwelle liegt, exponentiell abnimmt. Abbildung 7 illustriert diesen Sachverhalt. Es ist zu beobachten, dass in den beiden großen Kollektionen der Prozentsatz sehr ähnlicher Dokumente klein ist, während im Plagiatkorpus ein deutlich größerer Anteil vorliegt.

Um dieser Verteilung Rechnung zu tragen und um die Ähnlichkeitshashfunktionen vergleichbar zu machen, wurden diese so parametrisiert, dass die durchschnittliche Anzahl zurückgegebener Dokumente pro Anfrage nicht zu groß und etwa gleich war. Hierfür wurde die Anzahl der verwendeten Fuzzifizierungsschemata bzw. Zufallsvektormengen angepasst (siehe Abschnitt 3.3): zwei bis drei Fuzzifizierungsschemata für Fuzzy-Fingerprinting, zwischen 10 und 20 Zufallsvektormengen für Locality-Sensitive-Hashing.

Abbildung 8 stellt den Recall beider Hashing-Ansätze in Abhängigkeit der Ähnlichkeitsschwellen für die drei Testkollektionen gegenüber. Bei hohen Ähnlichkeitsschwellen ($> 0.8$) ist der Recall bei beiden Hashing-Ansätzen ausgezeichnet. Ein hoher Recall für *niedrige* Ähnlichkeitsschwellen ist hingegen nur zufällig erreichbar, was an der linken und mittleren Kurve für die RCV1- und die Web-Kollektion zu beobachten ist. Dieses Verhalten lässt sich durch die Verteilung der Ähnlichkeiten in Abbildung 7 erklären. Beide Hashing-Ansätze verhalten sich ähnlich, wobei Fuzzy-Fingerprinting einen leicht besseren Recall erzielt. Bei dem Plagiatkorpus hingegen zeigt sich ein anderes Bild: Fuzzy-Fingerprinting übertrifft Locality-Sensitive-Hashing deutlich bei der Erkennung sehr ähnlicher Dokumente.

Abbildung 9 stellt die Precision beider Hashing-Ansätze in Abhängigkeit von den Ähnlichkeitsschwellen für die drei Testkollektionen gegenüber. Offensichtlich ist – unabhängig von der Testkollektion – die Precision von Fuzzy-Fingerprinting signifikant höher als die von Locality-Sensitive-Hashing. Das heißt, eine von Fuzzy-Fingerprinting für eine Anfrage zurückgegebene Ergebnismenge $D'$ enthält entweder mehr relevante Dokumente oder sie ist kleiner als die von Locality-Sensitive-Hashing gelieferte Ergebnismenge. Beides hat direkten Einfluss auf die Suchzeit pro Anfrage und die Ergebnisqualität.

Die Laufzeit exakter Retrieval-Ansätze, die beispielsweise auf dem Vektorraummodell basieren, ist linear in der Größe der Dokumentkollektion. Im Gegensatz dazu kann die Laufzeit der Hashing-Ansätze als konstant betrachtet werden. Das Verhältnis der Precision-Kurven in Abbildung 9 gibt Aufschluss über das Verhältnis dieser Konstanten. Insbesondere ließ sich in den Experimenten beobachten, dass die durchschnittliche Größe $|D'|$ der Ergebnismenge pro Suchanfrage linear mit der Größe $|D|$ der Kollektion steigt, falls Art und Verteilung der Dokumente

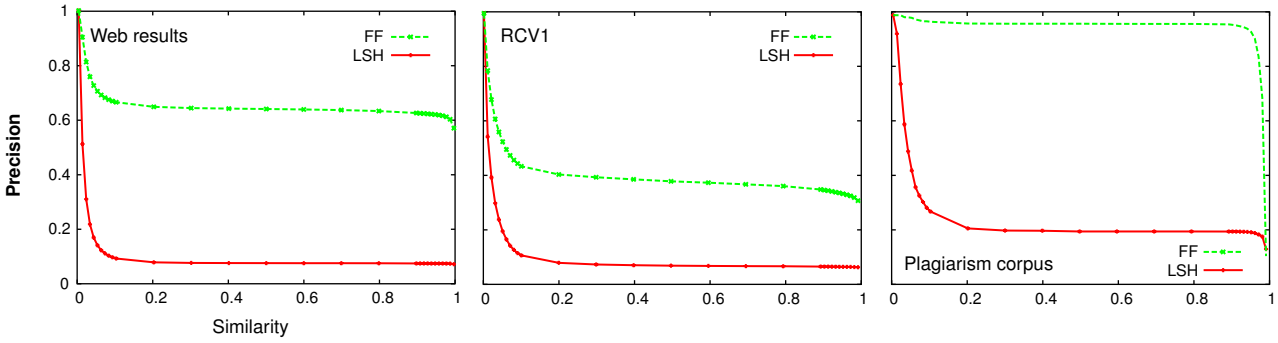


Abbildung 9: Die mit Fuzzy-Fingerprinting (FF) und Locality-Sensitive-Hashing (LSH) auf den drei Testkollektionen erzielte Precision in Abhängigkeit von der Ähnlichkeit.

der Kollektion sich nicht ändern.

Die Precision von Fuzzy-Fingerprinting wird durch die Anzahl $k$ von Präfixäquivalenzklassen und die Anzahl $r$ von Abweichungsintervallen je Fuzzifizierungsschemata gesteuert. Um die Precision zu erhöhen, genügt die Vergrößerung von einem der beiden Parameter. Der optimale Wert für $k$ ist von der Retrieval-Aufgabe abhängig; typische Werte für $r$ liegen zwischen zwei und vier. Die Precision von Locality-Sensitive-Hashing steigt mit der Anzahl $k$ der verknüpften Hashfunktionen. Bei Verwendung der Hashfamilie von Datar *et al.* entspricht $k$ der Anzahl der Zufallsvektoren je Hashfunktion; typische Werte für $k$ liegen zwischen 20 und 100.

### 4.3 Diskussion

Die Ergebnisse der Experimente geben einen Überblick über das Verhalten hashing-basierter Indizierung bei der Identifikation fast gleicher Dokumente und bei der Ähnlichkeitssuche.

Fuzzy-Fingerprinting ist Locality-Sensitive-Hashing bei der Ähnlichkeitssuche im Hinblick auf den Recall nur leicht voraus. Wir erklären diesen Sachverhalt mit dem geringen Anteil von Dokumenten in den Kollektionen, deren paarweise Ähnlichkeit größer als 0.5 ist. Bezüglich der Precision ist Fuzzy-Fingerprinting dem Verfahren des Locality-Sensitive-Hashing überlegen. Dieser Sachverhalt spielt jedoch nur eine untergeordnete Rolle, da der Umfang der Ergebnismenge einer Anfrage – verglichen mit der Größe der Dokumentkollektion – im Normalfall um Größenordnungen kleiner ist. Das heißt, Locality-Sensitive-Hashing kann für die Ähnlichkeitssuche genauso gut verwendet werden wie Fuzzy-Fingerprinting.

Bei der Identifikation sehr ähnlicher Dokumente hingegen übertrifft Fuzzy-Fingerprinting das Verfahren des Locality-Sensitive-Hashing sowohl bezüglich Precision als auch Recall. Für diese Art von Retrieval-Aufgaben ist Fuzzy-Fingerprinting fast konkurrenzlos.

## 5 Zusammenfassung

Hashing-basierte Indizierung ist eine vielversprechende Technologie im textbasierten Information-Retrieval, die zuverlässige und effiziente Anfragen nach ähnlichen Dokumenten für verschiedene Retrieval-Aufgaben gestattet. Wir haben drei wichtige Aufgabenklassen identifiziert, in denen hashing-basierte Indizierung Verbesserungspotenzial bietet, nämlich Gruppierung, Ähnlichkeitssuche und Klassifikation.

Es wurden zwei Konstruktionsprinzipien für Hashfunktionen vorgestellt: Fuzzy-Fingerprinting und Locality-Sensitive-Hashing. Eine umfassende experimentelle Analyse beider Hashing-Ansätze wurde durchgeführt, um (*i*) ihre Einsetzbarkeit bei der Ähnlichkeitssuche und der Suche nach fast identischen Dokumenten zu demonstrieren, und (*ii*) sie bezüglich Precision und Recall miteinander zu vergleichen.

Die Ergebnisse zeigen, dass Fuzzy-Fingerprinting bei der Suche nach fast identischen Dokumenten dem Verfahren des Locality-Sensitive-Hashing bezüglich Precision und Recall überlegen ist. Bei der Ähnlichkeitssuche ist die Precision von Fuzzy-Fingerprinting deutlich höher als die von Locality-Sensitive-Hashing, wohingegen nur ein leicht höherer Recall beobachtet wurde. Unsere Analyse beschränkte sich auf das textbasierte Information-Retrieval; wir möchten aber betonen, dass die vorgestellten Konzepte und Algorithmen auch für Retrieval-Aufgaben aus anderen Domänen adaptierbar sind. Insbesondere Locality-Sensitive-Hashing wurde daraufhin ausgelegt, unterschiedliche Arten hochdimensionaler, vektorbasierter Objektrepräsentationen zu verarbeiten. Auch die Prinzipien hinter Fuzzy-Fingerprinting sind übertragbar.

Unsere aktuellen Forschungen beschäftigen sich damit, hashing-basierte Indizierung theoretisch zu analysieren und die dabei gewonnenen Erkenntnisse in speziellen Retrieval-Aufgaben umzusetzen. Ziel ist es, die Beziehung zwischen den Determinanten von Fuzzy-Fingerprinting und der Retrieval-Performanz zu quantifizieren, um optimierte Hashindizes konstruieren zu können.

Fuzzy-Fingerprinting wird als Schlüsseltechnologie in unseren Werkzeugen zur textbasierten Plagiatanalyse eingesetzt.

## Literatur


Guy Aston und Lou Burnard. The BNC Handbook. `http://www.natcorp.ox.ac.uk/what/whatis.html`, 1998.

Ricardo Baeza-Yates und Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

Mayank Bawa, Tyson Condie und Prasanna Ganesan. Lsh forest: Self-tuning indexes for similarity search. In *WWW '05: Proceedings 14th international conference on World Wide Web*, S. 651-660, New York, NY, USA, 2005. ACM Press.

Andrei Z. Broder. Identifying and filtering near-duplicate documents. In *COM '00: Proceedings 11th Annual Symposium on Combinatorial Pattern Matching*, S. 1-10, London, 2000. Springer.

Moses S. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC '02: Proceedings 34th annual ACM symposium on Theory of computing*, S. 380-388, New York, NY, USA, 2002. ACM Press.

Mayur Datar, Nicole Immorlica, Piotr Indyk und Vahab S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *SCG '04: Proceedings 20th annual symposium on Computational geometry*, S. 253-262, New York, NY, USA, 2004. ACM Press.

Aristides Gionis, Piotr Indyk und Rajeev Motwani. Similarity Search in High Dimensions via Hashing. In *Proceedings 25th VLDB Conference Edinburgh, Scotland*, 1999.

Piotr Indyk und Rajeev Motwani. Approximate Nearest Neighbor—Towards Removing the Curse of Dimensionality. In *Proceedings 30th Symposium on Theory of Computing*, S. 604-613, 1998.

P. Indyk. Stable distributions, pseudorandom generators, embeddings and data stream computation. In *FOCS '00: Proceedings 41st Annual Symposium on Foundations of Computer Science*, S. 189, Washington, DC, USA, 2000. IEEE Computer Society.

Y. Matsuo und M. Ishizuka. Keyword extraction from a single document using word co-ocurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13(1):157-169, 2004.

Sven Meyer zu Eißen und Benno Stein. The AISEARCH Meta Search Engine Prototype. In Amit Basu und Soumitra Dutta, Eds., *Proceedings 12th Workshop on Information Technology and Systems (WITS 02), Barcelona*. Technische UniversitÃ?t Barcelona, Dezember 2002.

John P. Nolan. Stable distributions—models for heavy tailed data. http://academic2.american.edu/~jpnolan/stable/stable.html, 2005.

T.G. Rose, M. Stevenson und M. Whitehead. The Reuters Corpus Volume 1—From Yesterday's News to Tomorrow's Language Resources. In *Proceedings 3rd International Conference on Language Resources and Evaluation*, 2002.

Benno Stein und Sven Meyer zu Eißen. Near Similarity Search and Plagiarism Analysis. In M. Spiliopoulou, R. Kruse, C. Borgelt, A. Nürnberger und W. Gaul, Eds., *From Data and Information Analysis to Knowledge Engineering*, S. 430-437. Springer, 2006.

Benno Stein. Fuzzy-Fingerprints for Text-Based Information Retrieval. In Klaus Tochtermann und Hermann Maurer, Eds., *Proceedings 5th International Conference on Knowledge Management (I-KNOW 05), Graz*, Journal of Universal Computer Science, S. 572-579. Know-Center, July 2005.

Roger Weber, Hans-J. Schek und Stephen Blott. A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces. In *Proceedings 24th VLDB Conference New York, USA*, S. 194-205, 1998.

Wikipedia. Plagiarism. `http://de.wikipedia.org/wiki/Plagiat`, 2006.

Ian H. Witten, Alistair Moffat und Timothy C. Bell. *Managing gigabytes (2nd ed.): compressing and indexing documents and images*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

Oren Zamir und Oren Etzioni. Web Document Clustering: A Feasibility Demonstration. In *Proceedings 21st annual international ACM SIGIR conference on Research and development in information retrieval*, S. 46-54, University of Washington, Seattle, USA, 1998.