Institut für Formale Methoden der Informatik

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Masterarbeit Nr. 86

Resultatsclustering und einfache POI Exploration für OSCAR

Tobias Bagg

Studiengang:	Informatik
Prüfer/in:	Prof. Dr. Stefan Funke
Betreuer/in:	Prof. Dr. Stefan Funke, Dipl.-Inf. Daniel Bahrdt
Beginn am:	12. Oktober 2015
Beendet am:	7. April 2016
CR-Nummer:	I.5.3

Kurzfassung

Die von der Abteilung FMI/ALG entwickelte Suchmaschine „OSCAR“ (**OS**M **C**ell **AR**rangement [BF15]) erlaubt flexibles Suchen im weltweiten OpenStreetMap (OSM) Datenbestand. Hierbei werden sowohl Teilwort- und Präfixsuche auf Textinhalte, als auch alle üblichen Mengenoperationen und geometrischen Einschränkungen unterstützt. Während die zu Grunde liegende Suchmaschine bereits sehr weit entwickelt ist, gibt es bei der Bedienung beziehungsweise der grafischen Oberfläche von „OSCAR“ weiterhin Verbesserungspotenzial.

Bisher wird bei einer Suche, welche mehr als eine bestimmte Anzahl an Treffern liefert, lediglich eine Teilmenge auf der Karte dargestellt und der Benutzer kann bei Bedarf weitere Treffer nachladen. Dies führt vorerst zu einer unvollständigen Präsentation des Suchergebnis als auch zu Unübersichtlichkeit, falls die Suchtreffer geografisch sehr dicht liegen, da für jeden Treffer ein eigenes Kennzeichen auf der Karte gezeichnet wird. Hierbei wäre es wünschenswert, dem Nutzer die geografische Lage aller Suchtreffer präsentieren zu können ohne die gesamten Metainformationen der Treffer komplett auf den Rechner des Nutzers laden zu müssen. Dabei sollte ebenfalls die Präsentation der Resultate auf der Karte verbessert werden, indem geografisch dicht gelegene Resultate zusammengefasst werden (geografisches Clustering). Die Granularität der dargestellten Ergebnisse sollte hierbei abhängig vom Vergrößerungslevel der Karte gewählt werden und somit zu einem dynamischen Nachladen von Informationen führen.

Darüber hinaus kann die in den Kartendaten vorhandene Hierarchie von Gebieten genutzt werden, um dem Benutzer eine detaillierte Inspektion des Suchergebnis zu ermöglichen. Dies soll durch eine interaktive Visualisierung erreicht werden, wobei das Laden von Subregionen und Suchtreffern unterstützt wird.

Des Weiteren kann der Benutzer bei der Eingabe einer Suche unterstützt werden, um genauere Suchergebnisse zu erzielen. Um nach spezifischen Institutionen in den OSM-Daten suchen zu können, muss der Benutzer die zugehörigen OSM-Tags (Schlüssel-Wert Paare) zur Verfügung haben. Um beispielsweise nach Pizzerien zu suchen ist es nötig das OSM-Tag „@cuisine:pizza“ zu kennen. Hierfür kann dem Benutzer ein Menü angeboten werden, welches diese Tags bereitstellt.

Zusätzlich können Suchtreffer, insbesondere bei Suchen nach Restaurants, Parks oder interessanten Plätzen (sogenannte POI: „Point-of-Interest“) durch Bilder verbessert werden. OSM enthält hierfür keine eigenen Bilderdaten, jedoch kann untersucht werden, inwiefern öffentliche Bilderportale dazu geeignet sind Suchtreffer zu verbessern.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Listings

1 Motivation

OpenStreetMap (OSM) ist ein freies Projekt dessen Ziel die Kartografierung des Globus ist. Die Grundidee ist dabei, dass prinzipiell jeder die Kartendaten erweitern als auch auf diese zugreifen kann. Dies stellt einen erheblichen Vorteil gegenüber Produkten wie zum Beispiel „Google Maps“, „Microsoft Bing Maps“ oder „Here“ dar, da direkt und lizenzkostenfrei mit den Kartendaten gearbeitet werden kann, sowie keine Restriktion durch die Benutzereingabe des jeweiligen Herstellers vorhanden ist.

Durch die mittlerweile beachtliche Menge an Benutzern ist ein großer Bestand an Kartendaten angefallen. Zum Vergleich wuchs die Anzahl an OSM-Benutzerkonten vom 13. April 2009 zum 23. Oktober 2015 von knapp 100.000 auf circa 2,3 Millionen [Ope16]. Um die Kartendaten effektiv nutzen zu können wurde „OSCAR“ entwickelt, sodass Suchoperationen effizient (in Bezug auf Suchzeit und Speicherplatz) durchgeführt werden können. Damit die vorhandenen Kartendaten vollständig genutzt werden können ist es notwendig alle relevanten OSM-Tags (Schlüssel-Wert Paare) durchsuchen zu können. Durch die weiterhin schnell steigende Menge an Daten führt dies unausweichlich dazu, dass speicher- und zeiteffiziente Algorithmen sowie Datenstrukturen gewählt werden müssen, um diese Aufgabe bewältigen zu können [BF15].

Um die unterstützten Funktionalitäten von „OSCAR“ nutzen zu können ist eine einfache Bedienung sowie Visualisierung der Suchtreffer für den Benutzer notwendig. Bezüglich der Bedienung muss der Benutzer bei der Eingabe seiner Suche insofern unterstützt werden, dass er bei der Suche nach bestimmten Kategorien von Einrichtungen aus einer sortierten Menge die passende wählen kann. Beispielsweise bei der Suche nach Pizzerien muss über ein Menü das OSM-Tag „@cuisine:pizza“ der Suchanfrage angehängt werden können. Ebenfalls müssen Funktionen wie das Suchen innerhalb eines vom Benutzer definierten geometrischen Abschnitts, das Suchen entlang eines Pfads oder innerhalb des Bildschirmbereichs, auf verständliche Art in die grafische Oberfläche integriert sein.

Die Visualisierung der Suchtreffer grenzt sich in zwei Bereiche ab: zum einen müssen Suchtreffer abhängig vom Vergrößerungslevel der Karte geclustert auf der Karte dargestellt werden. Zum anderen kann die Hierarchie der OSM-Gebiete einer Suchanfrage mit Hilfe eines Graphen veranschaulicht werden. Dies ermöglicht dem Benutzer das Inspizieren der Verschachtelung von Gebieten, als auch zielgerichtetes Nachladen von Suchtreffern und Gebieten.

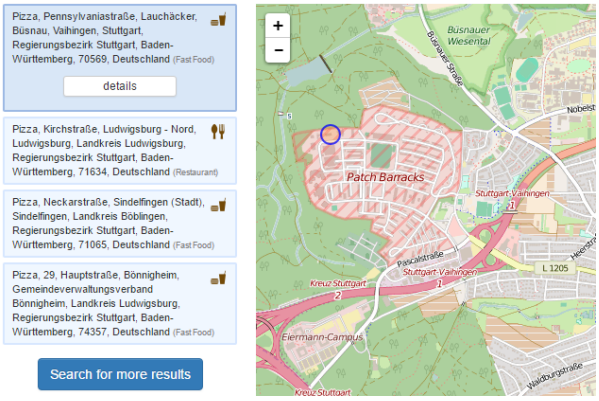


Abbildung 1.1: Suchergebnis von „Nominatim“ für die Suchanfrage „pizza Stuttgart“.

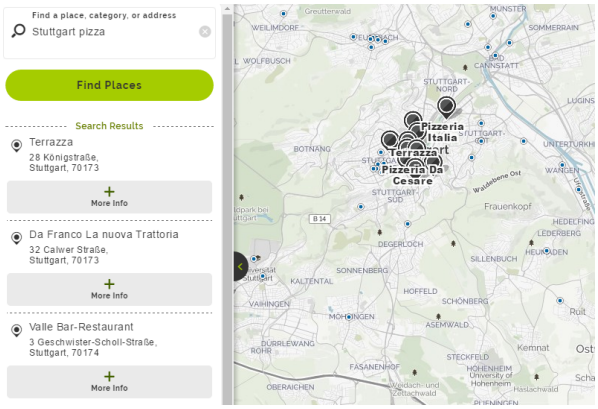


Abbildung 1.2: Suchergebnis von „MapQuest“ für die Suchanfrage „pizza Stuttgart“.

1.1 Verwandte Arbeiten

Projekte, welche ähnliche Zielsetzungen wie „OSCAR“ haben, wurden bereits in [BF15] aufgezählt und kurz analysiert. Innerhalb dieses Abschnitts soll deshalb spezifischer darauf eingegangen werden, inwiefern andere Projekte geclusterte Suchergebnisse erzeugen können.

Das offizielle OSM-Wiki listet im Artikel zu Suchmaschinen[1] mehrere Suchmaschinendienste auf, welche als Kandidaten zur Untersuchung betrachtet werden können. Hierzu gehört beispielsweise die offizielle OSM-Suchmaschine „Nominatim“[2], welche teilweise Basis dieser Dienste ist, als auch „MapQuest“[3].

Bei keinem der gelisteten Dienste konnte durch Testen mehrerer Suchanfragen eine Fähigkeit zum Clustern von Suchergebnissen festgestellt werden. Hierfür wurden Anfragen gewählt, welche eine relativ große Ergebnismenge erzeugen. Das Ergebnis der Suchanfrage „pizza Stuttgart“ für „Nominatim“ und „MapQuest“ sind in Abbildung 1.1 und 1.2 dargestellt. Eine oftmals angewandte Methode, als Alternative zum Clustering, ist einzelne Suchtreffer dem Benutzer zu präsentieren und zusätzliche Treffer bei Bedarf nachzuladen. Ob diese Treffer zufällig geladen oder mit Hilfe einer Bewertungsfunktion bestimmt werden ist dabei nicht ersichtlich geworden. Mit dieser Methode wird allerdings dem Benutzer das vollständige Suchergebnis vorenthalten.

Auch proprietäre Dienste, welche nicht auf OpenStreetMap basieren wie zum Beispiel „Google Maps“, bieten keine geclusterten Suchergebnisse. Diese wurden zusätzlich in Abschnitt 3.1.4 betrachtet und mit „OSCAR“ verglichen.

[1] http://wiki.openstreetmap.org/w/index.php?title=Search_engines&oldid=1204846
[2] http://openstreetmap.org
[3] http://www.mapquest.com/

2 Theoretischer Hintergrund

Innerhalb dieses Kapitels sollen Themengebiete und Techniken erläutert werden, welche eine maßgebliche Rolle für das Verständnis dieser Ausarbeitung und innerhalb der Implementierung gespielt haben. Hierzu gehört der grundlegende Aufbau des OpenStreetMap Kartenmaterial als auch das Clustern von geografischen Elementen auf einer Karte.

2.1 OpenStreetMap (OSM)

Zum Modellieren einer Karte nutzt OSM drei Grundelemente [Ope15a]: Knoten (engl. „node“) [Ope15b], Wege (engl. „way“) [Ope15d] und Relationen (engl. „relation“) [Ope15c]. Mit Hilfe dieser Werkzeuge können kleine Elemente wie beispielsweise Parkbänke als auch ganze Gebiete, welche viele Elemente gruppieren, beschrieben werden. Zusätzlich können diese Grundelemente durch frei wählbare Zusatzinformationen erweitert werden (sogenannte „Tags“), um eine bessere Beschreibung beziehungsweise Klassifikation zu erreichen. Ein Tag ist ein Schlüssel-Wert Paar (Format = Schlüssel:Wert, zum Beispiel „@cuisine:pizza“). Diese werden genutzt, um die Merkmale eines Elements zu beschreiben oder dieses zu klassifizieren. Prinzipiell kann sowohl beim Schlüssel als auch beim Wert ein beliebiger Text (maximal 255 Zeichen) gewählt werden. Allerdings haben sich im OSM-Umfeld einige Konventionen durchgesetzt, wie Elemente korrekt mit einem Tag versehen werden.

2.1.1 Knoten

Ein Knoten [Ope15b] bestimmt einen spezifischen Punkt auf der Erdoberfläche, welcher mindestens mit einer eindeutigen ID und einer geografischen Position (Längen- und Breitengrad) beschrieben wird. Knoten werden zum einen dafür genutzt um Standorte, wie zum Beispiel Hotels, Restaurants oder Sonderziele, zu beschrieben. Zum anderen können mehrere Knoten

Listing 2.1 Repräsentation eines Knoten mit der ID = „123“, welcher ein Restaurant beschreibt.

```
<node id="123" lat="51.5" lon="-0.14">
  <tag k="amenity" v="restaurant"/>
</node>
```

genutzt werden, um einen Weg (siehe Abschnitt 2.1.2) zu bilden. In Listing 2.1 ist die Repräsentation eines Knoten beispielhaft dargestellt. Das darin enthaltene „tag“-Attribut beschreibt den Knoten als Restaurant.

2.1.2 Wege

Ein Weg [Ope15d] besteht aus mindestens zwei und maximal 2000 geordneten Knoten. Dieser ist entweder Teil einer Relation (siehe Abschnitt 2.1.3) oder mit einem Tag erweitert. Wege können genutzt werden um Straßen zu modellieren, aber auch um Gebiete darzustellen. In Listing 2.2 ist ein Weg beispielhaft dargestellt. Dieser besitzt die ID „123“ und enthält die Knoten „1“, „2“ und „3“. Zusätzlich ist dieser mit zwei Attributen erweitert, welche den Namen („A8“) und eine Klassifikation (highway=motorway) enthalten.

Listing 2.2 Repräsentation eines Wegs

```
<way id="123">
  <nd ref="1"/>
  <nd ref="2"/>
  <nd ref="3"/>
  <tag k="highway" v="motorway"/>
  <tag k="name" v="A8"/>
</way>
```

Offener Weg

Ein Weg, dessen erster und letzter Knoten unterschiedlich sind, wird als offener Weg bezeichnet. Oftmals werden Straßen, Zugstrecken oder Flüsse als offener Weg beschrieben.

Geschlossener Weg

Ein Weg, dessen erster und letzter Knoten identisch sind, wird als geschlossener Weg bezeichnet. Dadurch wird ein Polygon gebildet, welches innerhalb der Kartendaten als Gebiet (zugehöriges Tag: „area=yes“) interpretiert werden kann.

2.1.3 Relationen

Eine Relation [Ope15c] fasst Elemente zusammen und kann aus Knoten, Wegen und weiteren Relationen zusammengesetzt sein. Relationen werden dazu genutzt, um Beziehungen zwischen den enthaltenen Elementen darzustellen. Die genaue Bedeutung wird durch die enthaltenen Tags verdeutlicht. Ein Beispiel ist in Listing 2.3 dargestellt. Die Relation mit der ID = „123“ gruppiert insgesamt vier Elemente. Drei Knoten (ID „1“, „2“ und „3“) und einen Weg (ID „4“),

Listing 2.3 Repräsentation einer Busroute mit Hilfe einer Relation

```
<relation id="123">
  <member type="node" ref="1" role=""/>
  <member type="node" ref="2" role=""/>
  <member type="way" ref="3" role=""/>
  <member type="node" ref="4" role=""/>
  <tag k="name" v="Bus Linie 1"/>
  <tag k="network" v="VVS"/>
  <tag k="route" v="bus"/>
  <tag k="type" v="route"/>
</relation>
```

welche als Mitglieder („member“) aufgezählt werden. Eine genaue Beschreibung, welchen Zweck diese Relation erfüllt wird erneut über die mitgelieferten Tags gegeben. Hierbei handelt es sich um eine Busroute des „VVS“ mit dem Namen „Bus Linie 1“.

2.1.4 Induzierte Hierarchie

Durch die Möglichkeit Elemente zu verschachteln wird automatisch eine Hierarchie innerhalb der Kartendaten induziert. Dies lässt die Interpretation zu, dass ein Element alle OSM-Tags seiner Elternknoten erbt (wird von „OSCAR“ genutzt). Eine beispielhafte Anordnung von Relationen und eines Knoten ist in Abbildung 2.1 und dessen zugehöriger gerichteter azyklischer Graph (DAG) in Abbildung 2.2 dargestellt.

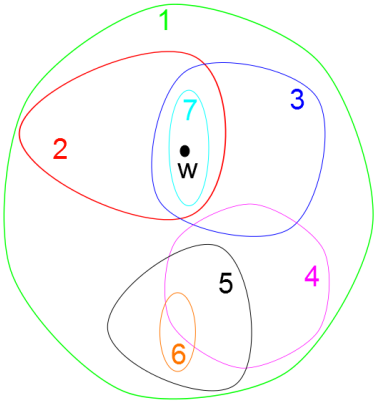


Abbildung 2.1: Mehrere Relationen (1-7) und ein Knoten „w“.

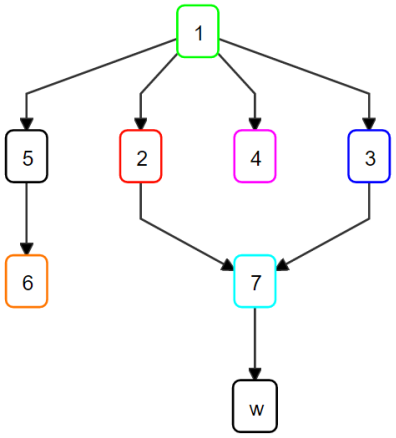


Abbildung 2.2: Zugehöriger gerichteter azyklischer Graph für Abbildung 2.1.

Eine Relation enthält entweder eine andere vollständig, überlappt sich mit dieser teilweise oder diese sind disjunkt. In OSM sind sogenannte Verwaltungsgrenzen (engl. „administrative

boundaries“) für gewöhnlich disjunkt und repräsentieren offiziell anerkannte Gebiete beziehungsweise Regionen wie zum Beispiel Länder, Bundesländer oder Bezirke. Es gibt allerdings auch Gebiete, welche sich mit anderen überlappen, wie zum Beispiel Übergangsgebiete von Verkehrszonen oder Parks.

Man erkennt in Abbildung 2.1 eine Relation „1“, welche alle anderen Relationen und den Knoten „w“ vollständig enthält. Deshalb stellt diese Relation die Wurzel im zugehörigen DAG in Abbildung 2.2 dar. Direkte Kinder der Wurzel sind die Relationen „2“, „3“, „4“ und „5“, nicht jedoch „6“, „7“ sowie der Knoten „w“. Die Relation „6“ ist ebenfalls vollständig in „5“ enthalten, sodass diese als Kind von „5“ eingeordnet wird. Relation „7“ ist sogar vollständig in „2“ und „3“ (welche beide Kinder von „1“ sind) enthalten und wird somit unter diesen beiden Knoten eingeordnet. Hier wird deutlich, dass ein Knoten mehrere Elternknoten besitzen kann. Der Knoten „w“ ist vollständig in Relation „7“ enthalten und wird somit dort als direkter Kindknoten eingeordnet.

Ein solcher DAG soll für eine Suchanfrage in der Implementierung visualisiert werden können. Dadurch kann der Benutzer genaue Informationen erlangen in welchen Regionen ein Suchtreffer enthalten ist.

2.2 Clustern von Datenelementen

Die Analyse von Datenmengen hat heutzutage einen hohen Stellenwert erhalten, insbesondere seitdem das Schlagwort „Big Data“ auch in der allgemeinen Presse thematisiert wird. Durch das Erheben von Daten in beinahe allen Lebensbereichen sollen diese für weiteren Erkenntnisgewinn verarbeitet werden, um beispielsweise bestimmte bisher versteckte Muster erkennen zu können. Das Gruppieren beziehungsweise Clustern von Daten ist eine Variante dieser Verarbeitung und ermöglicht es komplexe Datensätze durch Untergruppenbildung besser zu organisieren und somit verständlicher zu machen. Im Allgemeinen wird dabei versucht die Elemente einer Menge so zu trennen, dass möglichst homogene Gruppen entstehen [BJ81]. Hierbei spielt es zunächst keine Rolle nach welcher Regel diese Elemente einer Gruppe zugeordnet werden. Das Ziel der Gruppierung ist, dass Elemente, welche sich innerhalb der gleichen Gruppe befinden nach einer festgelegten Metrik mehr ähneln als Elemente aus verschiedenen Gruppen [BJ81]. Es gibt dabei keine einheitliche Definition einer Gruppe (Cluster) [ELL01], meist wird sich allerdings auf die interne Homogenität beziehungsweise die externe Abgrenzung bezogen [XW+05].

Hauptsächlich relevant für diese Arbeit ist das Clustern von geographischen Objekten. Datenelemente besitzen in diesem Fall ein Attribut, welches deren Position in einem Koordinatensystem beschreibt. In Abbildung 2.3 und Abbildung 2.4 sind Suchtreffer beispielhaft als blaue Punkte auf einer Karte eingezeichnet. Möchte man für diese Suchtreffer ein Clustering durchführen, stellt sich die Frage auf Basis welches Modells diese Unterteilung stattfinden soll, beziehungsweise welches Ergebnis überhaupt erwünscht ist. Mögliche Gruppierungen sind

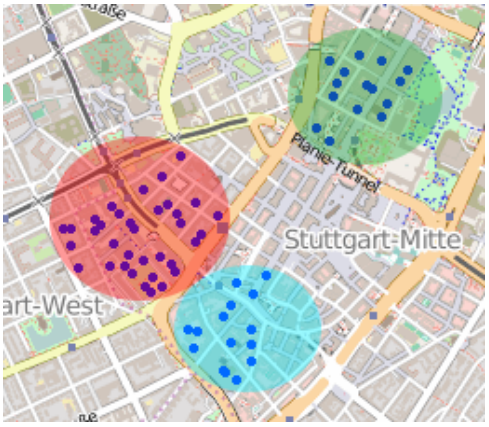


Abbildung 2.3: Darstellung von Suchtreffern als blaue Punkte auf einer Karte. Mögliches Clustering in drei Gruppen.

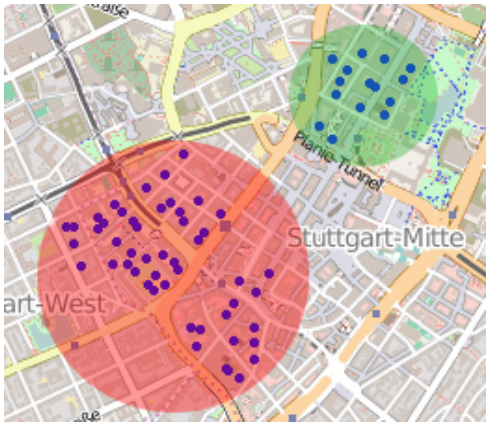


Abbildung 2.4: Darstellung von Suchtreffern als blaue Punkte auf einer Karte. Mögliches Clustering in zwei Gruppen.

direkt eingezeichnet: einmal wurden Suchtreffer in zwei (Abbildung 2.3) und das andere Mal in drei Gruppen (Abbildung 2.4) aufgeteilt. Hierbei zeigt sich eine der Schwierigkeiten beim Clustering, da beide Ergebnisse intuitiv als plausibel erscheinen. Das gewünschte Ergebnis bestimmt somit maßgeblich die Wahl der eingesetzten Clustermethode, beziehungsweise deren interne Parameter.

Generell kann man Clusteringmethoden in zwei größere Gruppen unterteilen: partitionierende und hierarchische Ansätze. Erstere teilen Datenobjekte direkt in Gruppen auf, während hierarchische Ansätze eine ganze Hierarchie von Gruppierungen generieren [XW+05]. Für die Implementierung sind hierarchische Verfahren interessant (für eine Begründung siehe Abschnitt 2.2.2) und sollen deshalb näher betrachtet werden.

2.2.1 Hierarchisches Clustering

Hierarchische Ansätze erzeugen eine hierarchische Struktur von Gruppierungen. Dies bedeutet, dass nicht nur ein Clustering berechnet wird, sondern mehrere unterschiedlicher Granularität. Diese Verfahren arbeiten dabei auf der Basis einer Distanz- beziehungsweise Nachbarschaftsmatrix [XW+05]. Diese symmetrische Matrix enthält zu Beginn in Zeile i und Spalte j das Ergebnis einer Distanzfunktion für die Elemente i und j eines Datensatz X [XW+05], welche im Ablauf des Algorithmus durch das Bilden von Clustern angepasst werden muss. Die Matrix wird genutzt, um zu bestimmen welche Elemente beziehungsweise Cluster im nächsten Schritt beispielsweise vereinigt werden.

Distanzfunktion

Eine Distanzfunktion muss in der Lage sein drei Fälle abzudecken [XW+05]:

- Ähnlichkeit zwischen zwei Elementen bestimmen
- Ähnlichkeit zwischen einem Element und einem Cluster bestimmen
- Ähnlichkeit zwischen zwei Clustern bestimmen

Basiert die Distanzfunktion zum Beispiel auf dem euklidischen Abstand, so muss eine zusätzliche Metrik eingeführt werden, um einen Cluster mit einem weiteren Cluster vergleichen zu können (ein einzelnes Element mit einem Cluster zu vergleichen verläuft analog). Hierfür gibt es unterschiedliche Möglichkeiten, zwei davon sind:

- Verwende das Ergebnis der Distanzfunktion des Paars mit dem geringsten Abstand („nearest neighbor) [XW+05]
- Verwende das Ergebnis der Distanzfunktion des Paars mit dem größten Abstand („farthest neighbor) [XW+05]

Agglomerierende und aufspaltende Methoden

Das hierarchische Clustering unterscheidet sich in zwei prinzipiellen Methoden: aufspaltende und agglomerierende Ansätze [XW+05]. Aufspaltende Ansätze arbeiten „Top-down“, sodass von einer Gruppe, die den gesamten Datensatz beschreibt, ausgegangen und diese schrittweise aufgespalten wird. Agglomerierende Ansätze arbeiten „Bottom-up“, beginnen mit einer Gruppe pro Element und fügen diese schrittweise zusammen bis lediglich eine Gruppe, welche ebenfalls die gesamten Elemente darstellt, übrig bleibt [XW+05].

Bei aufspaltenden Ansätzen stehen zu Beginn $O(2^N)$ Kombinationen der Aufspaltung zu Verfügung im Vergleich zu $O(N^2)$ Verschmelzungen bei agglomerierenden Verfahren. Dies bedeutet einen erheblichen Mehraufwand, sodass agglomerierende Verfahren in der Praxis häufiger Anwendung finden [XW+05].

Die Funktionsweise einer agglomerierenden Methode („nearest-neighbor“-Metrik für Cluster) soll im folgenden Beispiel erläutert werden, da diese innerhalb der Implementierung eingesetzt wurde. In Abbildung 2.5 ist ein Datensatz mit sieben Datenelementen in einem Koordinatensystem visualisiert. Die initiale Distanzmatrix auf Basis der Manhattan Distanz ($d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$) ist in Tabelle 2.1 aufgeführt. Der Algorithmus läuft schematisch wie folgt ab [XW+05]:

1. Initialisiere die Distanzmatrix und beginne mit N Clustern, die jeweils nur aus den Elementen des Datensatz bestehen.
2. Wähle das Paar aus der Distanzmatrix, welches die geringste Distanz besitzt und führe diese in einem Cluster zusammen.

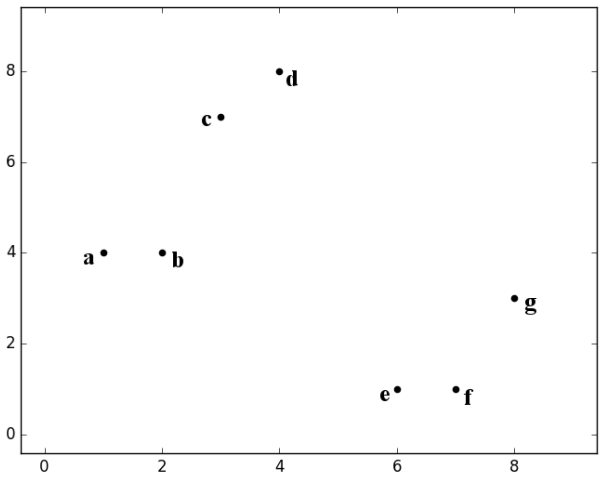


Abbildung 2.5: Sieben Datenpunkte, welche mit Hilfe einer hierarchischen Methode gruppiert werden sollen.

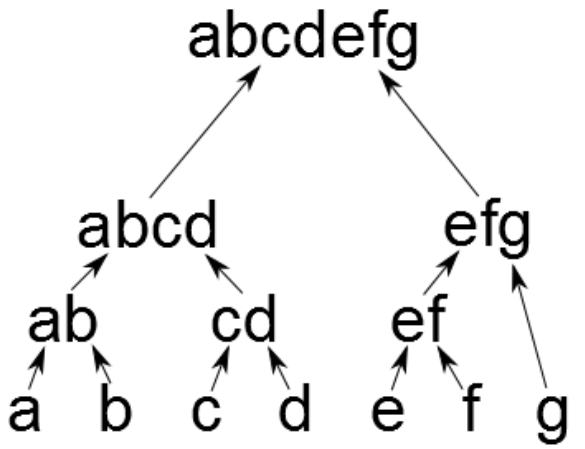


Abbildung 2.6: Dendrogram zu den in Abbildung 2.5 gruppierten Elementen.

	a	b	c	d	e	f	g
a	0	1	5	7	8	9	8
b	1	0	4	6	7	8	7
c	5	4	0	2	9	10	9
d	7	6	2	0	9	10	9
e	8	7	9	9	0	1	4
f	9	8	10	10	1	0	3
g	8	7	9	9	4	3	0

Tabelle 2.1: Initiale Distanzmatrix

	ab	c	d	e	f	g
ab	0	4	6	7	8	7
c	4	0	2	9	10	9
d	6	2	0	9	10	9
e	7	9	9	0	1	4
f	8	10	10	1	0	3
g	7	9	9	4	3	0

Tabelle 2.2: Distanzmatrix nachdem a und b zusammengefügt wurden

3. Passe die Distanzmatrix, entsprechend der Änderungen in 2., an.
4. Gehe zu 2., falls mehr als ein Cluster vorhanden ist.

Zu Beginn werden durch die vorliegende initiale Distanzmatrix in Tabelle 2.1 die Elemente „a“ und „b“ auf Grund des kleinsten Distanzwert „1“ (es könnte auch mit „ef“ oder „cd“ begonnen werden, welche den selben Wert besitzen) zu einem Cluster zusammengefügt. In der Matrix müssen darauf folgend alle Einträge, welche auf „a“ oder „b“ verweisen, angepasst werden. In diesem Fall wird jeweils der Distanzwert zu „b“ eingefügt, da „a“ zu jedem Datenelement eine höhere Distanz besitzt und „nearest-neighbor“ als Metrik für Cluster verwendet wird. Die daraus resultierende Matrix ist in Tabelle 2.2 dargestellt.

Anschließend werden die Elemente „c“ und „d“ mit Distanzwert „1“ vereinigt. Dies resultiert in der Matrix in Tabelle 2.3.

	ab	cd	e	f	g
ab	0	4	7	8	7
cd	4	0	9	10	9
e	7	9	0	1	4
f	8	10	1	0	3
g	7	9	4	3	0

Tabelle 2.3: Distanzmatrix nachdem c und d zusammengefügt wurden

Nun wird das letzte Paar von Elementen mit einem Distanzwert von „1“ vereint, nämlich „e“ und „f“. Dies führt zu der Matrix in Tabelle 2.4. Zu diesem Zeitpunkt befinden wir uns in der zweiten Hierarchiestufe von unten im Dendrogram in Abbildung 2.6.

	ab	cd	ef	g
ab	0	4	7	7
cd	4	0	9	9
ef	7	9	0	3
g	7	9	3	0

Tabelle 2.4: Distanzmatrix nachdem e und f zusammengefügt wurden

	ab	cd	efg
ab	0	4	7
cd	4	0	9
efg	7	9	0

Tabelle 2.5: Distanzmatrix nachdem ef und g zusammengefügt wurden

Es bleiben nun drei Cluster mit zwei Elementen („ab“, „cd“ und „ef“) und ein Cluster mit einem einzelnen Element („g“) zur weiteren Auswahl übrig. Mit kleinstem Distanzwert können „ef“ mit „g“ (Distanzwert = 3), wie in Tabelle 2.4 ersichtlich, vereinigt werden.

Daraufhin werden „ab“ und „cd“ mit Distanzwert „4“ (Tabelle 2.5) zu einem Cluster verschmolzen, die zugehörige resultierende Distanzmatrix befindet sich in Tabelle 2.6.

	abcd	efg
abcd	0	7
efg	7	0

Tabelle 2.6: Distanzmatrix nachdem ab und cd zusammengefügt wurden

Bevor der Algorithmus beendet ist, werden die beiden letzten Cluster „abcd“ und „efg“ zusammengesetzt. Die gesamte daraus entstandene Hierarchie des Ablaufs ist ebenfalls in Abbildung 2.6 zu erkennen.

2.2.2 Clustern von Suchtreffern

Intuitiv eignet sich ein hierarchischer Ansatz zur Gruppierung von Suchtreffern auf einer Karte aus mehreren Gründen:

- Es können die erzeugten Hierarchiestufen genutzt werden, um über die Granularität der Darstellung innerhalb der jeweiligen Vergrößerungsstufe auf der Karte zu entscheiden.
- Hierarchische Ansätze werden oftmals in der Kombination mit Distanzmetriken (zum Beispiel dem euklidischen Abstand) verwendet. Dies scheint im vorliegenden Anwendungsfall, bei dem geographisch nahe beieinanderliegende Treffer zusammengefügt werden sollen, eine passende Lösungsvariante zu sein.
- Einen hierarchischen Ansatz verfolgt bereits OpenStreetMap durch den Aufbau des Kartenmaterials und „OSCAR“ nutzt diese Hierarchie serverseitig, um Anfragen geclustert zu beantworten.
- Ein agglomerierender Ansatz wird einem aufspaltenden, wie bereits erläutert, aus Leistungsvorteilen vorgezogen.

3 Implementierung

In diesem Kapitel sollen alle Verbesserungsvorschläge, welche bereits in Kapitel 1 angedeutet wurden, in ihrer Umsetzung näher erläutert werden. Hierzu gehören das Clustern von Suchtreffern, die Visualisierung der Hierarchie von OSM-Relationen, die Erweiterung der grafischen Oberfläche sowie das Anbinden von öffentlichen Bilderdiensten zur Ergänzung von Suchtreffern.

3.1 Clustering

Um dem Benutzer auch bei Suchanfragen mit großen Ergebnismengen eine vollständige geographische Präsentation der Suchtreffer zu ermöglichen ist ein zweistufiges Clustering durch Server und Client nötig. Der Server kann dabei die vorhandene Hierarchie von Relationen, wie in Abschnitt 2.1.4 beschrieben nutzen, um Suchtreffer bereits geographisch geclustert zu übermitteln. Der Client kann diese Cluster beim Vergrößern eines Kartenausschnitts durch spezifische Anfragen an den Server verfeinern. Wird dabei für einen Cluster eine festgelegte Anzahl an enthaltenen Suchtreffern unterschritten, werden diese vom Server geladen und vom Client geographisch geclustert.

Zusätzlich sollen Cluster verschiedener Größe dem Benutzer unterschiedlich dargestellt werden. Um große Cluster (repräsentieren mehr als 100 Suchtreffer) visuell schneller zu erfassen, haben

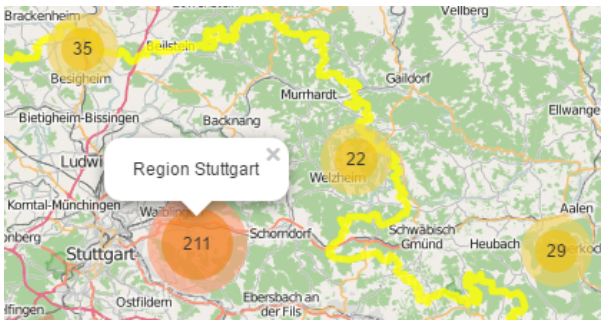


Abbildung 3.1: Ausschnitt des geclusterten Suchergebnis (durch den Server) für die Suchanfrage nach Pizzerien in Stuttgart.

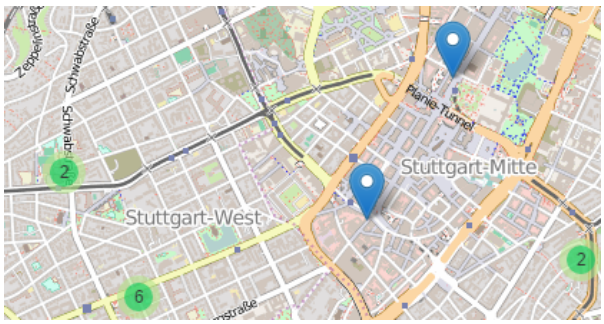


Abbildung 3.2: Vergrößerter Abschnitt aus Abbildung 3.1, einzelne Suchtreffer wurden durch den Client geclustert.

diese den größten Durchmesser und sind rot markiert (siehe Abbildung 3.1). Etwas kleiner und gelb gefärbt sind mittelgroße Cluster (repräsentieren mehr als 10, aber weniger als 100 Suchtreffer, siehe Abbildung 3.1). Die kleinsten Cluster sind grün dargestellt (weniger als 10 Suchtreffer, siehe Abbildung 3.2). Einzelne Suchtreffer bekommen ein eigenständiges Symbol, welches ebenfalls in Abbildung 3.2 dargestellt ist.

3.1.1 Server

Um große Suchergebnisse geclustert darstellen zu können ist es nötig, dass dieser Prozess teilweise schon auf dem Server stattfindet, damit der Client die benötigten Informationen nicht zuerst in vollem Umfang laden und verarbeiten muss. Diese Funktionalität ist bereits in „OSCAR“ vorhanden, sodass lediglich die Suchanfrage gestellt werden muss. Das gelieferte Ergebnis enthält bei großen Suchmengen zunächst nur die Relationen beziehungsweise Gebiete, in welchen die Treffer sich befinden. Durch spezifische Anfragen des Clients, zum Beispiel beim Vergrößern der Karte, werden diese dann weiter aufgelöst.

Ein Beispiel einer Suchanfrage („@cuisine:pizza Stuttgart“) ist in Abbildung 3.1 dargestellt. Beim Bewegen des Mauszeigers über einen Cluster wird zusätzlich der Name dargestellt, sowie die Grenzen des Clusters (in der die Suchtreffer liegen) in gelb auf die Karte gezeichnet.

3.1.2 Client

Um die Suchtreffer einer Anfrage vollständig und übersichtlich darstellen zu können, sollen geographisch nahegelegene Treffer zusätzlich geclustert werden. Hierfür bietet „leaflet“ [lea16a], eine Javascript Bibliothek für interaktive Karten, das Submodul „markercluster“ [lea16b], welches diesen Vorgang ermöglicht. Die genaue Funktionsweise soll im folgenden erläutert werden.

Das Modul „markercluster“ implementiert ein, wie bereits in Kapitel 2.2.1 erklärt, hierarchisches agglomerierendes Clusteringverfahren. Hierbei wird für jedes Vergrößerungslevel der Karte ein Clustering berechnet, wodurch unterschiedliche Granularitäten der Darstellung ermöglicht werden. Agglomerierende Verfahren müssen während des Clustering häufig entscheiden, welche Elemente als nächstes zusammengefügt werden. Wie in 2.2.1 beschrieben ist dies mit einer Distanzmatrix möglich, sodass immer das Paar mit der aktuell kleinsten Distanz verschmolzen wird. Diese Art der Implementierung benötigt relativ viel Speicherplatz für die benötigte Matrix. Zusätzlich muss in jedem Schritt die gesamte Matrix untersucht werden, um herauszufinden welches Paar zusammengefügt werden soll.

Listing 3.1 Pseudocode für ein Gitter als optimierte Alternative für eine Distanzmatrix

```
Gitter : {
  gitter: [][],
  hinzufügen(objekt): {
    i = abrunden(objekt.x / Gitterzellengröße);
    j = abrunden(objekt.y / Gitterzellengröße);
    gitter[i][j] += objekt;
  },
  nächstesObjekt(objekt): {
    i = abrunden(objekt.x / Gitterzellengröße);
    j = abrunden(objekt.y / Gitterzellengröße);
    nächstesObjekt = bestimme aus allen Nachbarzellen von gitter[i][j] das Element mit
        der geringsten Distanz zu "objekt"
    return nächstesObjekt;
  }
}
```

Verwendung von Gittern

Um keine Distanzmatrix einsetzen zu müssen wird als Optimierung pro Vergrößerungslevel ein Gitter (welches aus Zellen besteht) eingesetzt, welches mit fester Zellgröße (Standard: 80 Pixel x 80 Pixel) über die Karte gelegt wird. Dies ermöglicht es schnell ein anderes Element zu finden mit dem ein neuer Cluster gebildet werden oder einen bereits vorhandenen Cluster zu finden, welchem ein Element eingegliedert werden kann. Ermöglicht wird dies dadurch, dass der Suchraum verkleinert wird und nicht alle vorhanden Cluster beziehungsweise Elemente durchsucht werden müssen, sondern lediglich die in den Nachbarzellen des Gitters.

Der Pseudocode für ein Gitter ist in Listing 3.1 aufgeführt. Die zentrale Speicherstruktur ist ein mehrdimensionales Array („gitter“), welches als ersten Index eine Zeile und zweiten Index eine Spalte verwendet, um eine Zelle des Gitters eindeutig zu identifizieren. Die Funktion „hinzufügen()“ speichert ein Objekt, indem dessen Position im Gitter durch die Division der Objektkoordinate (auf der Karte) mit der Zellengröße berechnet wird. Außerdem zeigt die Funktion „nächstesObjekt()“ wie für ein Objekt aus dessen Nachbarschaft das Objekt mit der kleinsten Distanz bestimmt wird, indem alle Nachbarzellen, genauer gesagt die darin enthaltenen Elemente (insbesondere nicht alle vorhandenen Elemente auf der Karte) betrachtet werden.

Clustering von Objekten

Anwendung finden diese Gitter in Listing 3.2, in welchem das Clustering in Pseudocode dargestellt ist. Dort werden pro Vergrößerungslevel zwei Gitter gespeichert: eines für Cluster („clusterGitter“) und für einzelne Suchtreffer („markerGitter“). Weshalb dies nötig ist, wird im folgenden deutlich. Prinzipiell werden die Suchtreffer einzeln dem Clustering per Funktion

Listing 3.2 Pseudocode für die Clusteringmethode von leaflet.markercluster [lea16b]

```
Clustering: {
  clusterGitter: Gitter[], // jeweils pro Zoomlevel ein Gitter
  markerGitter : Gitter[], // jeweils pro Zoomlevel ein Gitter
  hinzufügen(objekt){
    für alle zoomLevel i >= maxZoom:
      if cluster = clusterGitter[i].nächstesObjekt(objekt):
        füge cluster neues Kind "objekt" hinzu;
        return;
      else if marker = markerGitter[i].nächstesObjekt(objekt):
        erzeuge neuen Cluster aus marker und objekt;
        return;
      else:
        füge objekt in markerGitter[i] ein;
  }
}
```

„hinzufügen()“ übergeben. Diese Funktion versucht, beginnend vom höchsten Vergrößerungslevel, das Element einem vorhandenen Cluster zuzuordnen beziehungsweise mit einem anderen Element einen neuen Cluster zu bilden. Ist dies möglich, endet die Funktion auf dem jeweiligen Level. Ist dies nicht möglich, zum Beispiel weil sich auch auf höchstem Vergrößerungslevel kein Element in der Nähe befindet um einen Cluster zu bilden, müssen weitere Vergrößerungslevel durchlaufen werden und das Element wird in der zugehörigen Zelle in „markerGitter“ platziert.

Angemerkt werden soll, dass dem Clustering nicht nur tatsächliche Suchtreffer übergeben werden, sondern auch Objekte welche bereits Cluster (zum Beispiel eine Region) darstellen und in dieser Form vom Server bereitgestellt werden. Diese werden ebenfalls in das Clustering einbezogen, da Regionen sich überlappen können. Würde man für solche Elemente direkt ein Element auf der Karte platzieren, könnten diese ungünstig nahe beieinander auf der Karte positioniert sein.

3.1.3 Dynamisches Verfeinern von Clustern

Wird auf eine Suchanfrage bereits eine geclusterte Antwort vom Server übermittelt, wie in Abbildung 3.1, ist bisher noch nicht erläutert worden auf Basis welcher Ereignisse weitere Daten zu diesen Clustern angefordert werden sollen. Beginnt der Benutzer beispielsweise damit die Karte zu vergrößern oder das Sichtfeld durch Verschieben der Karte zu verändern, so muss zu diesem Zeitpunkt festgestellt werden für welche Cluster es sinnvoll ist weitere Daten über deren Subhierarchie anzufordern und für welche nicht. Dabei gilt es, so viele Daten wie nötig und so wenige wie möglich anzufordern, um möglichst wenig Datenverkehr zu verursachen und dem Nutzer eine gute Bedienbarkeit mit kleinen Wartezeiten zu gewährleisten.

Berechnung der Überlappung

Für jeden Cluster übermittelt der Server eine detaillierte Grenze (siehe Abbildung 3.1 in gelb eingezeichnet), sowie einen rechteckigen Rahmen (engl. „bounding box“), welcher diese umschließt. Beide Repräsentationen bestimmen, in welchem Bereich sich die Suchtreffer eines Cluster befinden. Um über die Verfeinerung eines Clusters zu bestimmen kann dieser Rahmen genutzt werden, indem die prozentuale Überlappung des Rahmens mit dem Sichtfeld des Bildschirms berechnet wird. Wenn dieser einen konfigurierten Schwellwert überschreitet, wird mit Hilfe des Servers die Subhierarchie geladen. Empirisch hat sich hierbei ein Schwellwert von 40% als sinnvoll ergeben.

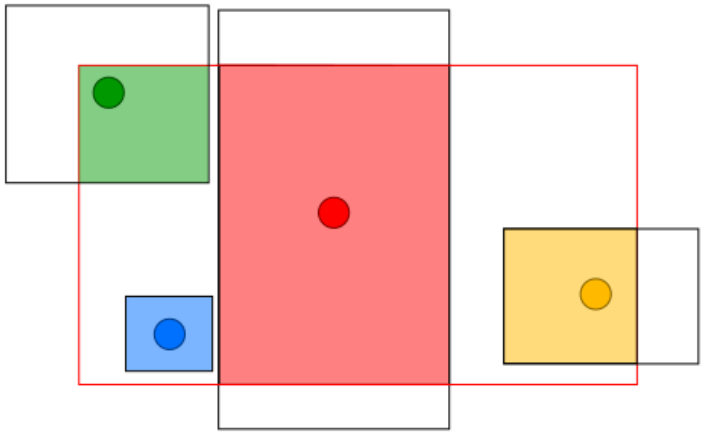


Abbildung 3.3: Darstellung von vier Clustern, sowie deren Rahmen. Der Bildschirm ist als rotes Rechteck markiert. Die Überlappung eines Clusterrahmen mit dem Bildschirm ist jeweils farbig eingezeichnet.

Der Vorgang ist in Abildung 3.3 visualisiert. Der Bildschirmbereich ist als rotes Rechteck eingezeichnet, sowie vier Cluster und deren Rahmen. Berechnet werden muss pro Cluster die Fläche der Überlappung des zugehörigen Rahmen mit dem Bildschirmbereich. Die überlappte Fläche wird dann durch die des Bildschirmbereichs dividiert, um den prozentualen Anteil zu erhalten. Die überlappte Fläche wird dabei nicht mit Hilfe der GPS-Koordinaten des Rahmen berechnet, sondern diese werden zuerst mit Hilfe der „leaflet“-Funktion ’„project()“ in ein Pixelkoordinatensystem projiziert, welches eine einfache Berechnung ermöglicht.

Zeichnen der Cluster

Bei jeder Interaktion mit dem Server, bei der Informationen über OSM-Relationen ausgetauscht werden, modelliert der Client diese Hierarchie im Speicher (ein azyklischer gerichteter Graph). Dies wird aus zwei Gründen durchgeführt:

Listing 3.3 Pseudocode zum Zeichnen von Cluster abhängig vom Vergrößerungslevel.

```
zeichneCluster(knoten) {
  if (knoten.kinder) {
    for (kind in knoten.kinder) {
      if (überlappung(kartenAuschnitt, kindKnoten.rahmen) >= schwellwert) {
        zeichneCluster(kind);
      } else {
        Clustering.hinzufügen(knoten);
      }
    }
  } else {
    lade Subhierarchie vom Server;
  }
}
```

- Erstens übernimmt diese Datenstruktur die Form eines Cache. Bewegt sich der Benutzer auf einen Teil der Karte, auf dem er sich schon einmal befunden hat muss keine Kommunikation mit dem Server stattfinden, sondern es können Daten verwendet werden, welche sich bereits lokal im Speicher befinden.
- Bei jedem Verschieben, Vergrößern oder Verkleinern der Karte wird die Funktion „zeichneCluster()“ aufgerufen und dieser wird als Parameter die Wurzel des Graphen übergeben. Der Pseudocode ist in Listing 3.3 dargestellt. Die Funktion bestimmt, welche Cluster auf die Karte gezeichnet werden und entscheidet somit über die Granularität der Darstellung. Dies wird dadurch erreicht, dass alle Cluster beziehungsweise Suchtreffer bestimmt werden, die angezeigt werden sollen. Diese werden dann dem Clustering übergeben (siehe Listing 3.3).

Als Beispiel wird davon ausgegangen, dass eine globale Suchanfrage nach Pizzerien durchgeführt wurde. Der Benutzer hat begonnen sich auf der Karte zu bewegen und Abschnitte zu vergrößern (es wurden dadurch mehrfach Aufrufe von „zeichneCluster“ durchgeführt und es hat Kommunikation mit dem Server stattgefunden). Nun wird erneut die Funktion („zeichneCluster(Welt)“) durch Interaktion des Benutzers aufgerufen. Der Bildschirmabschnitt befindet sich geographisch über der Region Stuttgart. Die Traversierung des Graphen ist in Abbildung 3.4 verdeutlicht. Auf Knoten die eine durchgezogene Kante zeigt wird die Funktion „zeichneCluster()“ rekursiv fortgesetzt (Überlappung mit dessen Rahmen ist hoch genug), bei gestrichelten Kanten ist dies nicht der Fall. Der Schwellwert zur Verfeinerung eines Knoten befindet sich bei 40% und die jeweilige prozentuale Überlappung eines Knoten mit dem Bildschirmausschnitt ist in Abbildung 3.4 zu finden.

Wie in Listing 3.3 beschrieben, werden die Kinder des Knoten „Welt“ („zeichneCluster(Welt)“) iteriert. Dabei ist lediglich die Überlappung mit dem Knoten „Deutschland“ größer als die Variable „Schwellwert“. Bei „Frankreich“ und „Schweiz“ ist dies nicht der Fall, sodass lediglich deren Cluster auf die Karte gezeichnet werden und die Rekursion für diese Knoten beendet wird.

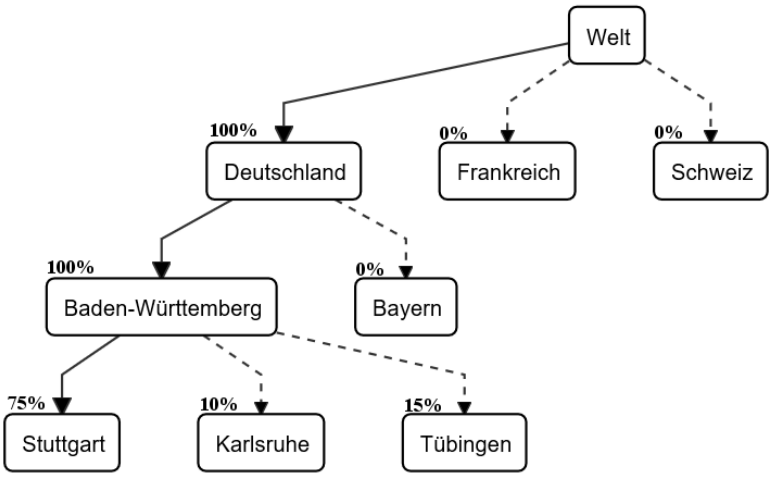


Abbildung 3.4: Verdeutlichung wie der Graph im Speicher traversiert wird. Knoten am Ende einer durchzogenen Kante werden rekursiv verfeinert (Rahmenüberlappung höher als Schwellwert). Für andere Knoten wird ein Clusterelement auf der Karte platziert. Die Überlappung mit dem Bildschirmbereich ist beispielhaft prozentual pro Knoten angegeben, der Schwellwert zur Verfeinerung liegt bei 40%.

Für „Deutschland“ wird die Rekursion per „zeichneCluster(Deutschland)“ fortgesetzt. Aus dessen Kindknoten erzeugt „Baden-Württemberg“ eine Überlappung, welche groß genug ist, um dort die Rekursion fortzusetzen. „Bayern“ wird dabei als Cluster auf die Karte gezeichnet. Der Aufruf „zeichneCluster(Baden-Württemberg)“ führt dazu, dass „Karlsruhe“ und „Tübingen“ als Cluster auf die Karte gezeichnet werden.

Die Rekursion wird dann am Knoten „Stuttgart“ per „zeichneCluster(Stuttgart)“ fortgesetzt, da hier allerdings keine Subhierarchie in Form von Kinderknoten im Graphen vorhanden ist und in diesem Beispiel die Überlappung dieses Knoten höher ist als der Schwellwert, wird in diesem Fall der Server kontaktiert und diese geladen. Dadurch muss ebenfalls die Graphstruktur im Speicher modifiziert werden, sodass der Knoten „Stuttgart“ weitere Kindknoten erhält.

Wie beschrieben muss die Funktion „zeichneCluster()“ bei jeder Interaktion des Benutzers (Verschiebung, Vergrößerung oder Verkleinerung der Karte) aufgerufen werden. Dabei werden alle darzustellenden Cluster und Suchtreffer eingesammelt und dem Clustering übergeben. Dieser Vorgang muss insgesamt schnell durchgeführt werden, da es sonst für den Benutzer zu lästigen Verzögerungen auf der Karte kommt. Durch die beschriebene Traversierung der OSM-Hierarchie ist dies möglich.

Abbildung 3.5: Dargestellte Suchtreffer in „Google Maps“ im Kartenausschnitt der USA für Suchanfrage "Pizza".

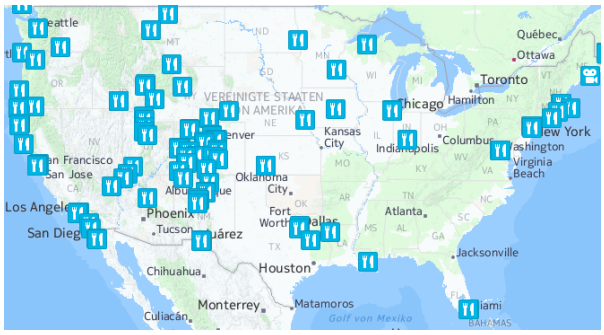


Abbildung 3.6: Dargestellte Suchtreffer in „Here“ im Kartenausschnitt der USA für Suchanfrage "Pizza".

Abbildung 3.7: Dargestellte Suchtreffer in „Bing“ im Kartenausschnitt der USA für Suchanfrage "Pizza".

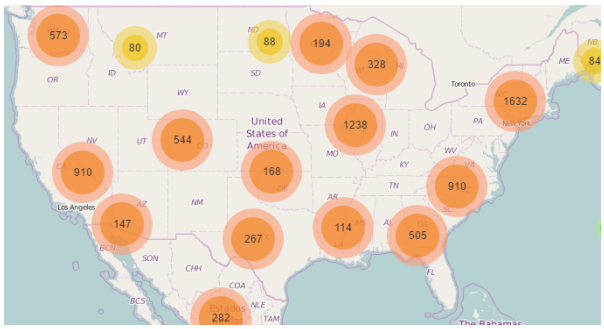


Abbildung 3.8: Dargestellte Suchtreffer in „OSCAR“ im Kartenausschnitt der USA für Suchanfrage „@cuisine:pizza“.

3.1.4 Evaluation des Clustering

Eine wichtige Eigenschaft, welche „OSCAR“ umsetzt, ist Suchanfragen dem Benutzer vollständig auf der Karte zu präsentieren. Dies ermöglicht es dem Benutzer genau festzustellen, in welchen Teilen der Karte wie viele Suchtreffer zu erwarten sind. Diese Funktionalität wird durch das bereits vorhandene Clustering durch „OSCAR“ und das lokale Clustering durch den Client, welches in dieser Arbeit hinzugefügt wurde, ermöglicht. Diese Funktionalität soll mit etablierten Suchdiensten verglichen werden.

In den Abbildungen 3.5, 3.6, 3.7, 3.8 sind die Kartenausschnitte der USA für die Suchanfrage „Pizza“ beziehungsweise „@cuisine:pizza“ für „OSCAR“ dargestellt. In Abbildung 3.5 ist das Suchergebnis von „Google Maps“ dargestellt. Genau wie bei „Here“ (Abbildung 3.6), werden einzelne Suchtreffer für die USA geladen und auf der Karte eingezeichnet. Für das jeweilige Vergrößerungslevel der Karte ist allerdings nicht ersichtlich, wie viele Treffer tatsächlich in den USA vorhanden sind und die dargestellte Menge scheint unvollständig zu sein. Beim

Vergrößern der Karte werden weitere Suchtreffer spezifisch für den gewählten Teil der Karte nachgeladen. Bei „Bing“ werden keine weiteren Suchtreffer geladen, weder beim Verschieben noch beim Vergrößern der Karte. Wie man in Abbildung 3.7 erkennen kann wurden statisch einige wenige Treffer für die USA dargestellt (eventuell auf Basis einer Bewertungsfunktion).

Generell scheint keiner der Dienste dem Nutzer eine vollständige Präsentation des Suchergebnis (bei Suchanfragen mit relativ großer Ergebnismenge) zu ermöglichen. Zum einen werden nicht alle Treffer dargestellt und die vorhanden werden einzeln, nicht geographisch gruppiert, auf die Karte gezeichnet. Zum Vergleich ist in Abbildung 3.8 das präsentierte Ergebnis durch „OSCAR“ dargestellt. Darin erkennt man wie das Clustering für große Suchanfragen eingesetzt wird und dem Nutzer direkt verfügbar ist in welchen Bereichen der USA welche Anzahl an Treffern vorhanden ist.

3.2 Grafische Oberfläche

Um eine einfache und ansprechende Oberfläche für den Benutzer anbieten zu können wurde ein am Rand der Karte positioniertes Menü (engl. „sidebar“) [Bie16] verwendet, wie in Abbildung 3.9 zu erkennen ist. Dieses kann zu jedem Zeitpunkt verkleinert werden, sodass der volle Blick auf die Karte möglich ist. Die wichtigsten integrierten Funktionen (in Abbildung 3.9 rot markiert und mit einer Zahl versehen) sollen im folgenden erläutert werden.

Mit einer roten „1“ sind Icons markiert, welche es ermöglichen zwischen den Untermenüs zu wechseln. Dazu gehören das Suchmenü, wie in Abbildung 3.9 zu sehen. Zweitens das Hilfesystem (Abbildung 3.11), welches zusätzliche Hilfe für die Eingabe von fortgeschrittenen Suchanfragen gibt als auch beispielhafte Suchen auflistet. Und Drittens ein Optionsmenü, welches dem Benutzer ermöglichen soll einzelne Funktionen der Oberfläche benutzerdefiniert zu steuern. Hier kann beispielsweise das Laden von Bildern vom Bilderdienst „flickr“ zu- und abgeschaltet werden.

Die rote „2“ markiert einen Schalter mit dessen Hilfe zwischen globaler und lokaler Suche gewechselt werden kann. Der globale Modus setzt keine zusätzliche geographische Einschränkung, außer diese welche bereits in der textlichen Suchanfrage enthalten sind. Der lokale Modus ermöglicht es lediglich im aktuellen Bildschirmbereich zu suchen.

Mit „3“ ist eine Verbesserung des Suchtext integriert. Hierbei werden Teilworte des Suchtexts in zusammenhängende Zeichen (engl. „token“) zerlegt. Dies hat zwei Vorteile: erstens können schnell Teile des Suchtext per einmaligem Mausklick entfernt werden. Zweitens können die OSM-Tags, welche verwirrend auf Benutzer wirken, versteckt werden. Das zusammenhängende Zeichen „pizza“ im Suchfeld beinhaltet das OSM-Tag „@cuisine:pizza“ und wird erst beim Abschicken der Suche in dieses übersetzt.

Die rote „4“ zeigt einen Link, welcher beim Klicken ein in dieser Abbildung verstecktes Menü für weitere Suchoptionen bietet. Darin enthalten sind Funktionalitäten um eine Suchanfrage

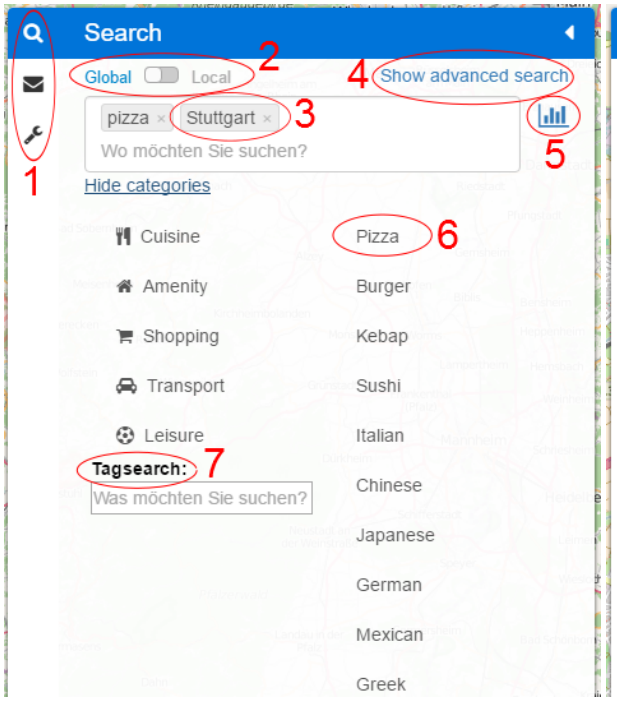


Abbildung 3.9: Am Rand der Karte positioniertes Menü [Bie16], welches eine einfach Interaktion mit „OSCAR“ ermöglicht.

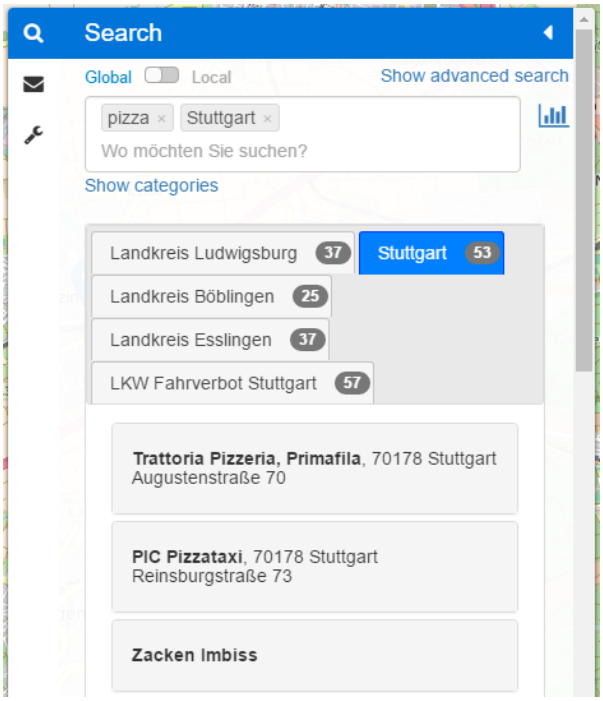


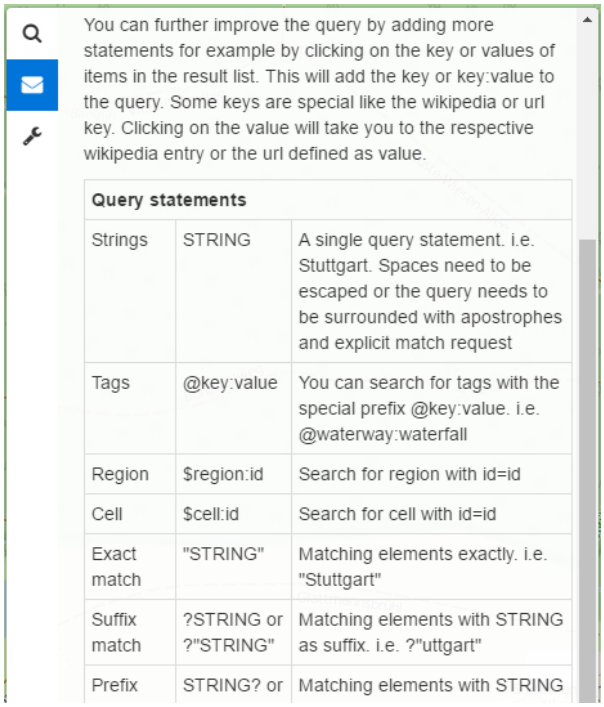
Abbildung 3.10: Suchtreffer werden getrennt nach Regionen (aus denen diese geladen wurden) in Tabs eingeordnet.

geographisch einzuschränken, beispielsweise durch das Definieren eines Polygons auf der Karte oder eines Pfads entlang dessen gesucht werden soll.

„5“ öffnet per Klick die Visualisierung der OSM-Hierarchie für die aktuelle Suche. Auf diese Option wird in Abschnitt 3.3 näher eingegangen.

Zusätzlich zeigt „6“ das bereits in Kapitel 1 angedeutete Menü für OSM-Tags. In der linken Spalte sind Kategorien wie beispielsweise zur Suche nach Restaurants („Cuisine“) aufgelistet. Durch das Klicken auf eine dieser Kategorien öffnet sich in einer weiteren Spalte deren Unterkategorien, zum Beispiel „Pizza“ oder „Burger“. Wird auf eine dieser Unterkategorien geklickt wird automatisch eine zusammenhängende Zeichenkette in das Suchfeld eingefügt, welche das OSM-Tag weiterhin vor dem Benutzer versteckt und erst beim Abschicken der Suche übersetzt wird.

Abschließend zum Menü, welches mit „6“ markiert ist, gibt es eine OSM-Tag Suche, welche mit „7“ hervorgehoben wird. Möchte der Benutzer eine Suche starten, für welche kein passender Eintrag im Menü (Abbildung 3.9, rote „6“) vorhanden ist, kann die dort integrierte Suche genutzt werden, um ein passendes „Tag“ zu finden. In Abbildung 3.12 ist diese Suche zu sehen,



You can further improve the query by adding more statements for example by clicking on the key or values of items in the result list. This will add the key or key:value to the query. Some keys are special like the wikipedia or url key. Clicking on the value will take you to the respective wikipedia entry or the url defined as value.

Query statements		
Strings	STRING	A single query statement. i.e. Stuttgart. Spaces need to be escaped or the query needs to be surrounded with apostrophes and explicit match request
Tags	@key:value	You can search for tags with the special prefix @key:value. i.e. @waterway:waterfall
Region	$region:id	Search for region with id=id
Cell	$cell:id	Search for cell with id=id
Exact match	"STRING"	Matching elements exactly. i.e. "Stuttgart"
Suffix match	?STRING or ?"STRING"	Matching elements with STRING as suffix. i.e. ?"uttgart"
Prefix	STRING? or	Matching elements with STRING

Abbildung 3.11: Das Hilfesystem gibt Auskunft über die möglichen Operatoren für eine Suchanfrage.

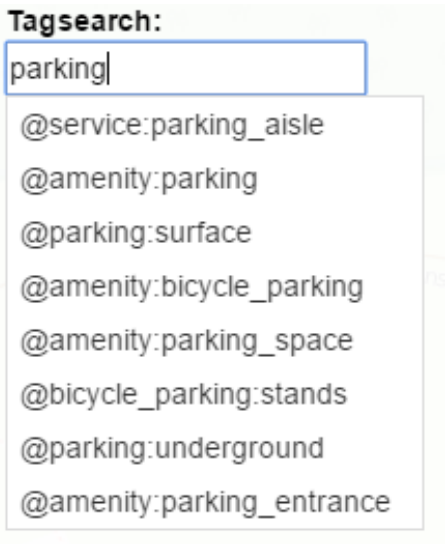


Abbildung 3.12: Falls der Benutzer nach einer spezifischen Einrichtung suchen möchte, welche im vorgesehenen Menü nicht enthalten ist, kann dieser versuchen das zugehörige OSM-Tag über diese Suche zu finden.

wobei dort nach einem „Tag“ für Parkplätze gesucht wird. Hierfür wird die Schnittstelle von „taginfo“[1] verwendet, welche eine umfangreiche Datenbank für OSM-Tags bereitstellt.

In der Abbildungen 3.11 ist ein Ausschnitt des Hilfesystem dargestellt. Dieses listet detailliert die unterstützten Operationen für Suchanfragen auf und gibt Beispiele an, welche diese verwenden.

3.3 Visualisierung der OSM-Hierarchie

Um die in Abschnitt 2.1.3 beschriebene Hierarchie von OSM-Relationen (Gebiete, Bezirke, Länder, etc.) zu visualisieren, wurde „dagre-d3“ [Pet16], eine Bibliothek zum Visualisieren von gerichteten Graphen, genutzt.

In Abbildung 3.13 ist der Graph für die Suchanfrage „@cuisine:pizza Stutgart“ dargestellt. Dies ermöglicht dem Benutzer zusätzlich zu erfahren wie die Gebiete heißen in denen sich

[1] http://taginfo.openstreetmap.org/

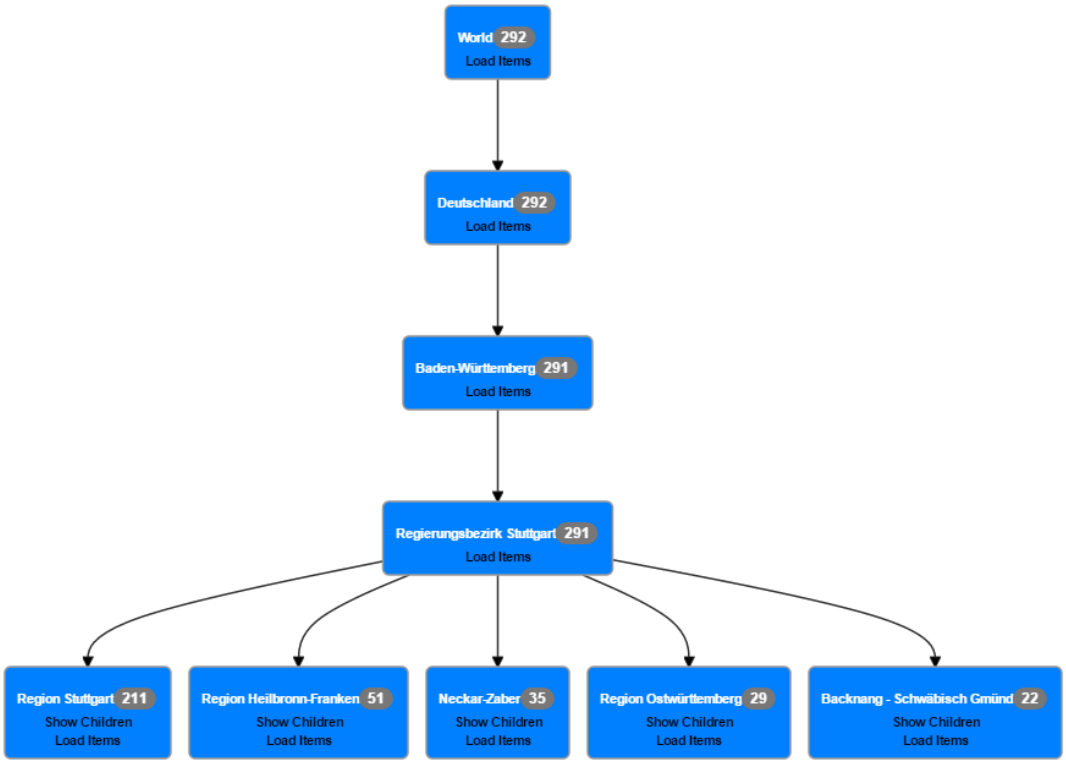


Abbildung 3.13: Visualisierte Hierachie der OSM-Relationen für die Suchanfrage „@cuisine:pizza Stuttgart“

die einzelnen Suchtreffer befinden sowie wie viele Treffer dort vorhanden sind. Der Graph ermöglicht es pro Knoten die direkte Subhierarchie oder die enthaltenen Suchtreffer vom Server zu laden. Gezeichnet wird diese Repräsentation mit Hilfe von „5“ (rot markiert) in Abbildung 3.9, wobei sich ein jederzeit schließbares Fenster über der Karte öffnet. Dieses enthält zusätzlich zwei Funktionen, um mit dem Graph zu interagieren.

- Es kann der gesamte Graph geladen werden (falls dieser nicht zu groß ist beziehungsweise zu viele Knoten besitzt).
- Es kann ein Modus aktiviert werden, welcher beim Nachladen einer Subhierarchie den Graph kompaktifiziert. Dies wird dadurch erreicht, dass ein Pfad vom geklickten Knoten zur Wurzel gesucht wird und nur Knoten auf dem Pfad, sowie deren Geschwister dargestellt werden. Dies ermöglicht eine übersichtlichere Darstellung bei großen Graphen.

3.4 Anbindung an öffentliche Bilderdienste

OpenStreetMap bietet keine integrierte Möglichkeit, um die von „OSCAR“ gelieferten Suchtreffer mit Bildern zu verknüpfen. Gleichzeitig sind Bilder eine Möglichkeit, um Suchtreffer maßgeblich zu verbessern. Ein Benutzer, der zum Beispiel auf der Suche nach einem Restaurant ist, könnte an Hand der Bilder (des Restaurants oder der angebotenen Speisen) entscheiden wo er gerne essen möchte. In den letzten Jahren sind viele Dienste, wie beispielsweise soziale Netzwerke oder Anbieter von kostenlosem Onlinespeicher für Bilder, entstanden. Diese bieten oftmals zusätzlich eine Schnittstelle, um automatisiert veröffentlichtes Material zu durchsuchen und auf diese Bilder zuzugreifen.

Einer dieser Dienstleister ist „flickr“[2], dessen Schnittstelle [fli16] sich als passend für das Ergänzen von Suchtreffern ergeben hat und unter einer festen URL erreichbar ist. Zusätzlich müssen mehrere Parameter für eine Anfrage gesetzt werden, um passende Bilder zu einer Suche zu erhalten.

- Parameter „lat“ und „lon“: GPS-Koordinate, welche das Suchgebiet geographisch einschränken soll.
- Parameter „text“: Suchtext, zum Beispiel der Name des Restaurant

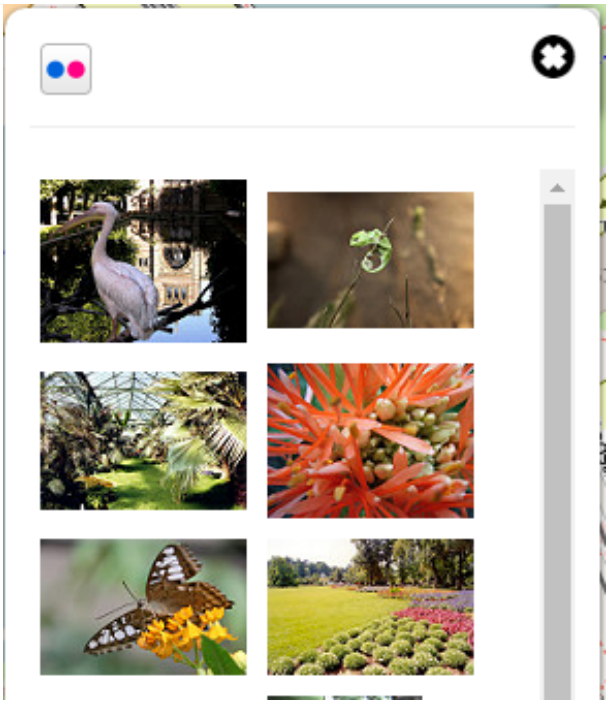


Abbildung 3.14: Ausschnitt der verfügbaren Bilder zur Suchanfrage „Stuttgart Wilhelma“

[2] https://www.flickr.com/

Sind für einen Suchtreffer Bilder vorhanden werden diese in einem zusätzlichen schließbaren Bereich am Bildschirmrand dargestellt. In Abbildung 3.14 sind vorgeschlagene Bilder für die Suchanfrage „Stuttgart Wilhelma“ dargestellt. Beim Klicken eines Bilds öffnet sich eine höher aufgelöste Version. Die gesamte Funktionalität kann vom Benutzer abgeschaltet werden, falls dieser keine Bilder angezeigt bekommen möchte.

Empirisch zeigt sich, dass einige Kategorien von Suchanfragen die Ergänzung mit Bildern besser ermöglichen als andere. Hierzu gehören Suchanfragen zu Lokalitäten an dem sich Menschen in ihrer Freizeit aufhalten, wie beispielsweise Bars, Restaurant oder Parks. Dort nehmen Personen vorzugsweise Bilder auf und stellen diese ins Internet.

4 Zusammenfassung

Innerhalb dieser Abschlussarbeit wurde die Benutzerschnittstelle der OpenStreetMap Suchmaschine „OSCAR“ überarbeitet und erweitert. Hierfür wurde eine Methode implementiert, um dem Benutzer jede Suchanfrage vollständig auf einer Karte zu visualisieren, unabhängig von der Anzahl der Suchtreffer. Die grafische Oberfläche wurde insofern angepasst, dass sie bei Bedarf vollständigen visuellen Zugriff auf die Karte zulässt, als auch den Benutzer bei der Eingabe einer Suche unterstützt. Um eine möglichst präzise Suchanfrage formulieren zu können, ohne dass dafür technisches Verständnis über OpenStreetMap notwendig ist, wurde ein Menü eingebunden, welches das Übersetzen von Suchbegriffen in OSM-Tags übernimmt. Zusätzlich lässt sich eine Suchanfrage mit Hilfe eines Graphen visualisieren, dessen Knoten OSM-Relationen wie zum Beispiel Länder, Regionen oder Gebiete repräsentieren. Dieser ist interaktiv gestaltet, sodass zielgerichtet weitere Hierarchieebenen oder Suchtreffer (die auf der Karte geclustert dargestellt sind) nachgeladen werden können. Außerdem wurde eine Anbindung an den Bilderdienst „flickr“ hergestellt, welcher die Suchtreffer einiger Kategorien (beispielsweise Restaurants, Hotels oder bekannte Plätze) von Suchanfragen mit Hilfe von Bildmaterial aufwerten kann.

4.1 Vergleich mit der alten Benutzerschnittstelle

In den Abbildungen 4.1 und 4.2 sind die alte (Abbildung 4.1) sowie die neue (Abbildungen 4.2) Benutzerschnittstelle im Vergleich dargestellt.

Hier wird deutlich wie das Clustering eingesetzt wird, um dem Benutzer eine Suchanfrage vollständig auf der Karte zu präsentieren. Im Vergleich wurde bei der alten Version lediglich eine Teilmenge der Treffer vom Client geladen und als einzelne Punkte auf die Karte gezeichnet. Bei Bedarf konnte der Benutzer weitere Suchtreffer durch das Scrollen der Trefferliste nachladen.

Die Suchleiste, Auflistung von Suchtreffern sowie ein Menü zur Unterstützung der Sucheingabe wurden kompakt in ein neues seitlich positioniertes Menü verlagert. Die alte Darstellung der OSM-Hierarchie wurde wie in Abschnitt 3.3 beschrieben durch einen interaktiven Graphen ersetzt. Innerhalb der Suchleiste wurde zusätzlich durch die Integration von zusammenhängenden Zeichen („Token“) erreicht, dass OSM-Tags vor dem Benutzer versteckt werden und die Suchanfrage leicht bearbeitet werden kann.

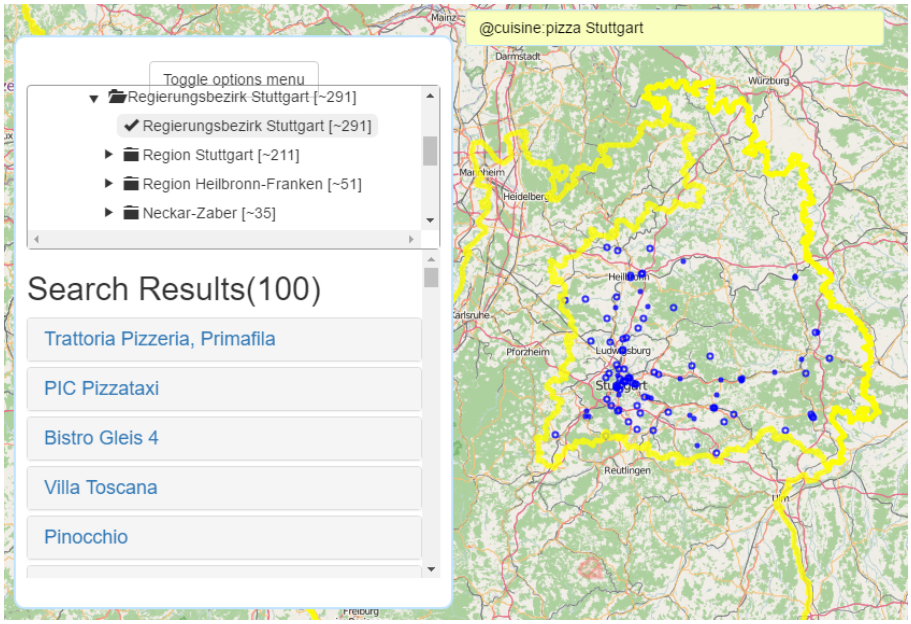


Abbildung 4.1: Dargestellt ist die alte Benutzerschnittstelle, welche kein Clustering bei Suchanfragen durchführt. Es wird lediglich eine Teilmenge der Suchtreffer vom Client geladen.

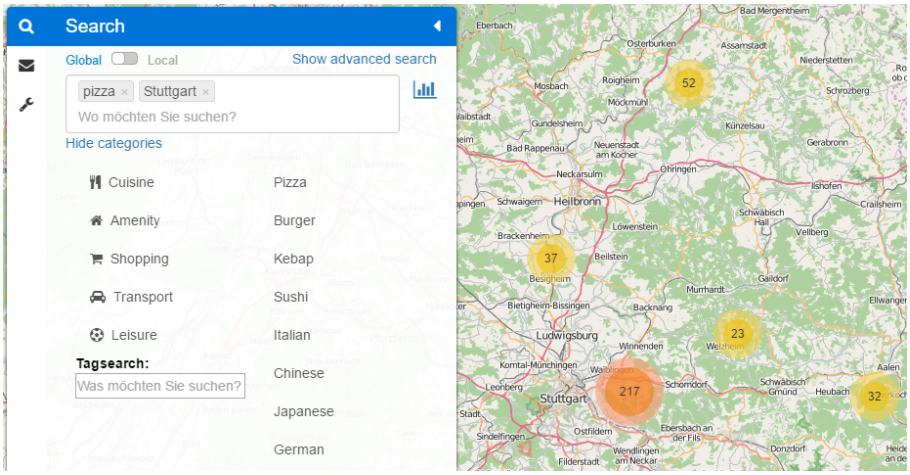


Abbildung 4.2: Dargestellt ist die neue Benutzerschnittstelle, welche ein Clustering der Suchtreffer durchführt und dem Benutzer somit die Position aller Treffer auf der Karte präsentieren kann.

4.2 Ausblick

Folgende Punkte bieten weitere Möglichkeiten zur Verbesserung, um insbesondere die Benutzbarkeit zu steigern.

Falsch platzierte Cluster durch große OSM-Rahmen

Aktuell sind im Kartenmaterial Länder wie zum Beispiel die USA, Neuseeland oder Frankreich vorhanden, die einen sehr großen umschließenden Rahmen besitzen. Dies resultiert meist aus mehreren Landteilen, die mehrere tausend Kilometer entfernt sind. Dadurch wird zum einen das Clusterelement falsch platziert (aktuell in der Mitte des Rahmen) und befindet sich zusätzlich meist an verwirrenden Positionen wie über einem Ozean. Zum anderen werden Cluster zu schnell aufgelöst, obwohl dies nicht nötig wäre. Eine Lösung wäre hierbei mehrere Rahmen für die einzelnen Landabschnitte einzuführen.

Breite des visualisierten azyklischen gerichteten Graph

Globale Suchanfragen wie zum Beispiel „@cuisine:pizza“ erzeugen eine unübersichtliche Suchgraph Visualisierung. Die Wurzel „Welt“ dieses Graphen besitzt beispielsweise alle Länder als Kinder, in denen es Suchtreffer für Pizzerien gibt. Dies wäre dadurch zu Lösen, dass eine weitere Hierarchieebene in den Kartendaten eingeführt wird, sodass Länder mit Hilfe der Kontinente gruppiert werden.

Kompression von Gebietsgrenzen

Beim Bewegen des Mauszeigers über ein Cluster wird die detaillierte Grenze der repräsentierten Region auf der Karte angezeigt. Insbesondere für große Ländergrenzen, wie zum Beispiel der USA, müssen hierfür relativ große Datenmengen übertragen werden. Dies kann je nach Datenverbindung des Benutzers zu langen Ladezeiten führen. Eine Möglichkeit diese Daten zu komprimieren ist einen Algorithmus zur Kantenglättung, wie beispielsweise den „Douglas-Peucker“-Algorithmus, auf die Grenzen anzuwenden.

Überlappende Gebiete

Offiziell anerkannte Gebiete, sogenannte „administrative boundaries“, überlappen sich für gewöhnlich nicht, sodass deren dargestellte Clusterelemente auf der Karte disjunkt sind. Für Gebiete wie Parks oder Übergangsgebiete von Verkehrszonen gilt dies allerdings nicht. Dadurch kommt es teilweise zu einer verwirrenden Darstellung, da eventuell Cluster, welche sich Suchtreffer teilen, gleichzeitig dargestellt werden. Beim Auflösen dieser Cluster bekommt

der Benutzer den Eindruck, dass nicht alle Suchtreffer dargestellt sind, da weniger Treffer angezeigt werden als zuvor durch die Cluster angepriesen.

Quellen

[BF15] D. Bahrdt, S. Funke. „OSCAR: OpenStreetMap Planet at Your Fingertips via OSm Cell ARrangements“. In: *Web Information Systems Engineering–WISE 2015*. Springer, 2015, S. 153–168 (zitiert auf S. 3, 13, 14).

[Bie16] T. Bieniek. *leaflet-sidebar*. [Online; accessed 6-April-2016]. 2016. URL: https://github.com/Turbo87/sidebar-v2 (zitiert auf S. 33, 34).

[BJ81] E. Backer, A. K. Jain. „A clustering performance measure based on fuzzy set decomposition“. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 1 (1981), S. 66–75 (zitiert auf S. 18).

[ELL01] B. Everitt, S. Landau, M. Leese. „Cluster Analysis Arnold“. In: *A member of the Hodder Headline Group, London* (2001) (zitiert auf S. 18).

[fli16] flickr. *flickr-API flickr.photos.search*. [Online; accessed 6-April-2016]. 2016. URL: https://www.flickr.com/services/api/flickr.photos.search.html (zitiert auf S. 37).

[lea16a] leaflet. *leaflet interactive maps*. [Online; accessed 6-April-2016]. 2016. URL: http://leafletjs.com/ (zitiert auf S. 26).

[lea16b] leaflet. *leaflet markercluster*. [Online; accessed 6-April-2016]. 2016. URL: https://github.com/Leaflet/Leaflet.markercluster (zitiert auf S. 11, 26, 28).

[Ope15a] OpenStreetMap. *OpenStreetMap Elemente*. [Online; accessed 6-April-2016]. 2015. URL: https://wiki.openstreetmap.org/w/index.php?title=Elements&oldid=1227835 (zitiert auf S. 15).

[Ope15b] OpenStreetMap. *OpenStreetMap Knoten*. [Online; accessed 6-April-2016]. 2015. URL: https://wiki.openstreetmap.org/w/index.php?title=Node&oldid=1189910 (zitiert auf S. 15).

[Ope15c] OpenStreetMap. *OpenStreetMap Relationen*. [Online; accessed 6-April-2016]. 2015. URL: https://wiki.openstreetmap.org/w/index.php?title=Relation&oldid=1249238 (zitiert auf S. 15, 16).

[Ope15d] OpenStreetMap. *OpenStreetMap Wege*. [Online; accessed 6-April-2016]. 2015. URL: https://wiki.openstreetmap.org/w/index.php?title=Way&oldid=1267511 (zitiert auf S. 15, 16).

[Ope16] OpenStreetMap. *OpenStreetMap Statistik*. [Online; accessed 6-April-2016]. 2016. URL: http://wiki.openstreetmap.org/w/index.php?title=Stats&oldid=1231181 (zitiert auf S. 13).

[Pet16] C. Pettitt. *dagre-d3*. [Online; accessed 6-April-2016]. 2016. URL: https://github.com/cpettitt/dagre-d3 (zitiert auf S. 35).

[XW+05] R. Xu, D. Wunsch et al. „Survey of clustering algorithms“. In: *Neural Networks, IEEE Transactions on* 16.3 (2005), S. 645–678 (zitiert auf S. 18–20).

Alle URLs wurden zuletzt am 06. 04. 2016 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift