# Verkehrscharakterisierung durch Methoden des maschinellen Lernens

Benjamin Wiesmüller
Seminar Innovative Internet-Technologien und Mobilkommunikation, WS 2008/2009
Institut für Informatik, Lehrstuhl Netzarchitekturen und Netzdienste
Technische Universität München
benny.w@mytum.de

## KURZFASSUNG

In dieser Arbeit werden Ergebnisse aktueller (2004-2006) Forschungsarbeiten zur Internet-Verkehrscharakterisierung mit Hilfe von Methoden des maschinellen Lernens vorgestellt. Zu Beginn folgt in der Einleitung ein kleiner Überblick über Anwendungsmöglichkeiten für Verkehrscharakterisierung und es werden bisher dafür verwendete Verfahren und deren Probleme vorgestellt. Anschließend wird einen Einstieg in das Thema maschinelles Lernen mit Definitionen und Überblick über verschiedene Methoden geboten, speziell im Hinblick auf die Verkehrscharakterisierung. Danach folgen Zusammenfassungen von drei konkreten Forschungsarbeiten zum Thema mit unterschiedlichen Verfahren zur Verkehrscharakterisierung mit:

- Clustering (Unsupervised Learning),
- Expectation Maximization (Unsupervised Learning),
- und Classification (Supervised Learning).

## Schlüsselworte

Verkehrscharakterisierung, Maschinelles Lernen, Clustering, Classification

## 1. EINLEITUNG

In dieser Arbeit, werden Ergebnisse aus aktuellen Forschungsarbeiten ([1],[2],[3],[4],[5]) zur Netzwerkverkehrs-charakterisierung zusammengefasst und einige der möglichen Anwendungen und verwendeten Techniken vorgestellt.

Zunächst was ist mit Verkehrscharakterisierung genau gemeint? Es sollen an einem Knotenpunkt, z.B. dem Internetzugang eines Unternehmens, die verschiedenen Verbindungen untersucht und anschließend automatisch kategorisiert werden. Beispielsweise könnte man sie zu einer konkreten Anwendung, oder einer bestimmten Verhaltensklasse zuordnen. Dies kann für Netzwerkadministratoren aus verschiedenen Gründen nützlich sein: Netzüberwachung, Optimierung der Netzwerkarchitektur, Analyse des aktuellen Verkehrs, Simulation für verschiedene Netzbelastungen, Quality of Serivce (Priorisierung von Verbindungen), Sicherheitsaspekte (Erkennen von verdächtigem Netzwerkverkehr), etc.

Bisher wurden hauptsächlich zwei Vorgehensweisen verwendet um den Verkehrsfluss zu charakterisieren: Analyse der Paket-Header und des Paket-Payloads. Aus dem Header eines Pakets kann man IP-Adresse und Portnummer, jeweils von Ziel und Quelle, sowie das verwendete Protokoll erkennen. Eine Möglichkeit eine Verbindung nun zu einer Anwendung zuzuordnen, ist mit Hilfe der Portnummer, über die von der IANA festgelegten Well-Known- oder anderweitig bekannten Ports von Anwendungen. Dies ist zwar einfach und schnell möglich, aber bei einigen neuen Anwendungen kaum mehr aussagekräftig [1]. Handelt ein Programm den verwendeten Port dynamisch aus, oder tunnelt ihren Daten gar über einen der Well-Known-Ports, so ist eine Erkennung anhand der Portnummern nicht mehr möglich. Die Analyse des Paket-Payloads kann dieses Problem umgehen, indem in der Nutzlast eines Pakets nach typischen Signaturen einer Anwendung gesucht wird. Jedes Paket so zu analysieren stellt aber hohe Anforderungen an Speicherbedarf und Rechenleistung, deswegen sind dafür auch spezielle Hardwarelösungen erhältlich. Außerdem stößt auch dieses Verfahren an seine Grenzen, wenn die Nutzlast verschlüsselt wird [1].

Die hier zugrunde liegenden Forschungsarbeiten untersuchten die Verwendung und Möglichkeiten von Methoden des maschinellen Lernens um diese Probleme zu umgehen und den Verkehr zuverlässig zu charakterisieren.

Die Arbeit ist im weiteren wie folgt eingeteilt: In Kapitel 2 folgt eine Überblick über maschinelles Lernen mit Hinblick auf die Verkehrsklassifizierung. In den nächsten drei Kapiteln werden die konkreten Verfahren vorgestellt. Kapitel 3 geht dabei über die Verwendung von Clustering (Unsupervised Learning) zur Zuordnung einer Verbindung zu einer Anwendung, mit den Algorithmen k-Means und DBSCAN. Kapitel 4 stellt ein Verfahren für eine schnelle Charakterisierung, nur anhand der ersten paar Pakete, vor, wobei die Expectation Maximization-Verfahren Gaussian Mixture Model, Hidden Markov Model und erneut k-Means verwendet werden. Kapitel 5 zeigt die Verwendung von Classification (Supervised Learning) um Verbindungen einigen vordefinierten Quality-of-Service-Klassen zuzuordnen. Dabei werden Nearest Neighbour und Linear Discriminant Analysis verwendet. Kapitel 6 schließt mit einem Fazit ab, das auf Möglichkeiten und Probleme der neuen Verfahren eingeht.

## 2. WAS IST MASCHINELLES LERNEN?

### 2.1 Definition

DIE festgelegte Definition von maschinellem Lernen gibt es nicht, deswegen findet man oft leicht unterschiedliche Definitionen dafür.

In [6] findet sich, wiederum geliehen von der englischen Wikipedia, übersetzt folgende Definition:

Maschinelles Lernen ist ein Untergebiet der künstlichen Intelligenz, die sich mit der Entwicklung von Algorithmen und Techniken beschäftigt, die dem Computer das „Lernen“ beibringen. Das induktive maschinelle Lernen extrahiert Regeln und Muster aus großen Datensätzen. Der Fokus der Forschung beim maschinellen Lernen liegt hauptsächlich darauf automatisch, mit rechnerischen und statistischen Methoden, Informationen aus Daten zu extrahieren.

Im Hinblick auf die Techniken die in dieser Arbeit erklärt werden, soll dass heißen, dass die Daten anhand verschiedener Attribute automatisch in verschiedene Kategorien/Gruppen eingeteilt werden sollen, indem Muster in diesen Attributen erkannt werden. So werden z.B. einzelne TCP-Verbindungen

anhand der Paketgrößen, Inter-Arrival-Time, Verbindungsdauer etc., zu den zugehörigen Anwendungen zugeordnet.

Üblicherweise werden beim maschinellen Lernen (mindestens) folgende drei Arten unterschieden [6]:

Überwachtes Lernen (Supervised Learning):

Für gegebene Eingaben werden dem lernenden System auch die dazugehörigen gewünschten Ausgaben geliefert. Das Ziel ist es korrekte Ausgaben für nicht weiter benannte Eingaben zu produzieren.

Hierbei sind die Eingaben also schon mit den zugehörigen Gruppenbezeichnungen versehen. Das System soll nun Muster erkennen anhand derer es zukünftige Eingaben unbekannter Zuordnung korrekt einordnen kann.

Unüberwachtes Lernen (Unsupervised Learning):

Hier ist das Ziel ein Modell aus den Eingaben zu erstellen ohne im Voraus zu wissen wie diese geartet sein sollen.

Das System soll hier anhand bestimmter Attribute der Daten diese zu ähnlichen Gruppen zusammen fassen. Hier wird häufig dem System die Anzahl der gewünschten Gruppen vorgegeben.

Bestärkendes Lernen (Reinforcement Learning):

Anstatt nur einfacher Ausgaben/Zuordnungen, sollen hier Aktionen produziert werden, die den Zustand der „Welt“ des Systems verändern. Je nach Aktion erhält das System „Belohnungen“ und es soll lernen diese zu maximieren (in dieser Arbeit nicht weiter von Bedeutung).

## 2.2 Unsupervised Learning

Beim Unsupervised Learning wird das Ermitteln der Gruppen, in die die Daten eingeteilt werden, Clustering genannt. Die entstehenden Gruppen entsprechend Cluster. Dies geschieht meist offline in einer Trainingsphase, bei der Trainingsdaten als Eingabe für den Algorithmus benötigt werden. Dazu werden sogenannte Traces verwendet. Dabei handelt es sich um Aufzeichnungen von Netzwerkverkehr, wobei Paket-Header-Traces, die nur die Header der IP-Pakete enthalten und Payload-Traces, die auch die Nutzlast enthalten, unterschieden werden. Um an solche Traces zu gelangen kann man z.B. frei erhältliche Aufzeichnungen des Internetverkehrs einiger Universitäten verwenden. Anschließend kann man die Traces mit diversen Tools analysieren, z.B. gibt es Programme die Payload-Traces analysieren können und so ermitteln zu welchen Anwendungen die jeweiligen Pakete gehören – mit entsprechendem Zeitaufwand. So kann man bestimmen welche Arten von Verbindungen sich in einem der entstandenen Cluster befinden. Wie schon in der Einleitung erwähnt stoßen solche Verfahren aber unter Umständen an ihre Grenzen, z.B. bei verschlüsselter Nutzlast. Eine mögliche Lösung in einem solchen Fall trotzdem geeignete Trainingsdaten zu erhalten, wäre manuell einen Trace zu erzeugen, z.B. den isolierten Verkehr einer einzelnen Anwendung aufzuzeichnen, und diesen als Eingabe zu verwenden [1].

Bevor man den Clustering-Algorithmus starten kann, muss man aus den Traces noch eine geeignete Repräsentation der Daten erzeugen. Wie schon angedeutet muss man sich dafür zunächst entscheiden welche Attribute der Daten betrachtet werden. Diese Auswahl unterscheidet sich je nach gewünschtem Ziel und wird in den jeweiligen Kapiteln zu den einzelnen Methoden erläutert.

Die spätere, meist online bei laufendem Verkehr stattfindende, Zuordnung von neuen Verbindungen zu den vorher entstandenen Clustern wird Classification genannt. Diese ergibt sich meist einfach aus den verwendeten Algorithmen, kann aber auch weitere Tricks verwenden um die Ergebnisse zu verbessern.

## 2.3 Supervised Learning

Beim Supervised Learning sind die Cluster und die Zuordnung der Eingabedaten zu diesen vorgegeben. Man spricht dann meist von Classification-Algorithmen. Nachteile solcher Verfahren sind, dass neue unbekannte Anwendungen nicht erkannt werden können [3]. Will man aber alle Eingaben von vornherein nur in eine feste Anzahl an Cluster einordnen, kann man diese durch ausgesuchte Eingabedaten charakterisieren, ohne genau zu wissen wie die Attributwerte der Eingaben genau geartet sind. Hier erkennt dann wieder der Algorithmus die Muster in den Werten. In Kapitel 5 werden die Eingabedaten z.B. in vier Verhaltensklassen eingeteilt, unabhängig von der Anwendung, die hinter den Verbindungen steckt.

Zur abschließenden Überprüfung kann wieder eine genaue Payload-Analyse der Ergebnisse benützt werden. So kann man z.B. die tatsächliche Anwendung einer Verbindung ermitteln und mit den Zuordnungen des neuen Verfahrens vergleichen. Dabei spricht man von True-Positives (TP) wenn eine Verbindung richtig klassifiziert worden ist und von False-Positives (FP) bei einer falschen Klassifikation.

## 2.4 Anwendungsgebiete

Übliche Anwendungsgebiete von maschinellem Lernen sind z.B. das Data-Mining, was manchmal auch synonym zu maschinellem Lernen verwendet wird. Data-Mining bezeichnet das Gewinnen von Wissen und Daten sowie dessen Darstellung und Anwendung. Die verwendeten Methoden kommen meist aus der Statistik oder der KI und sollen auch auf sehr große Datenmengen mit vertretbarem Aufwand anwendbar sein. Ein Beispiel dafür ist das Kundenempfehlungssystem bei Amazon, bei dem Ähnlichkeiten zwischen Produkten ermittelt werden und dem Kunden solche, die ähnlich zu von ihm gekauften sind, vorgeschlagen werden. Allgemein fällt benutzerspezifische Werbung in diesen Bereich. Auch bei Spam-Filtern für e-mails werden Techniken des maschinellen Lernens verwendet [7]. Ein weiteres Anwendungsgebiet ist Pattern Recognition, wie z.B. das automatische Erkennen von Gesichtern auf Bildern oder bei der Spracherkennung.

## 2.5 Verkehrsklassifizierung

Die Möglichkeit den Verkehr in einem Netzwerk analysieren und in verschiedene Kategorien, z.B. nach Anwendung oder Verhalten, einteilen zu können ist ein wichtiger Teil vieler Netzwerk-Management Aufgaben, wie die Priorisierung von bestimmten Verbindungen, Traffic-Shaping/Policing, diagnostische Überwachung etc. So könnte ein Netzwerk-administrator z.B. Verkehr von p2p-Anwendungen im Netz drosseln um genügend Leistung für wichtige Business-Anwendungen eines Unternehmens sicher zu stellen. Ähnlich dazu helfen obige Möglichkeiten auch bei netzwerktechnischen Problemstellungen wie Workload-Charakterisierung, Planung der benötigten Kapazität usw.

Im Folgenden werden einige Forschungsarbeiten vorgestellt, die versuchen mit Hilfe von Methoden des maschinellen Lernens Verfahren zu entwickeln, die zum Lösen der obigen Problemstellungen beitragen können.

# 3. CLUSTERING

In einer Arbeit von der University of Calgary in Kanada [3] wird versucht den Netzwerkverkehr zu verschiedenen Anwendungen zuzuordnen. Dabei werden Clustering-Techniken (Unsupervised Learning) und Statistiken aus der Transportschicht verwendet. Untersucht wurden hier zwei Algorithmen, nämlich k-Means und DBSCAN. Diese werden mit den Ergebnissen, des schon in einer anderen Arbeit untersuchten AutoClass-Algorithmus verglichen. Das Clustering beruht hier darauf, dass unterschiedliche Anwendungen unterschiedliches Verhalten im Netzwerk zeigen. So besitzt z.B. eine Dateiübertragung mit FTP über lange Verbindungszeiten und große durchschnittliche Paketgrößen, während ein Instant-Messaging Programm nur gelegentlich kleine Nachrichten abschickt.

Das Verfahren ist in zwei Schritte aufgeteilt. Zuerst einer Offline-Trainingsphase und anschließend einer Klassifikationsphase in der online oder offline der Netzwerkverkehr klassifiziert wird. Für die Trainingsphase werden zwei Traces von unterschiedlichen Universitätsnetzen verwendet (von Calgary und Auckland). Aus diesen werden die verschiedenen Verbindungen extrahiert und deren Attribute auf Transportlevelschicht untersucht. Dabei wurden hier die Gesamtzahl der Pakete, die mittlere Paketgröße, die mittlere Nutzdatengröße ohne Header, die Anzahl der übermittelten Bytes, jeweils in beide Richtungen und zusammen und die mittlere Inter-Arrival-Time der Pakete betrachtet.

## 3.1 k-Means

Eine Möglichkeit für das Clustering der Eingabedaten ist es sie in einem euklidischen Raum mit n Dimensionen zu betrachten. Wobei n der Anzahl der betrachteten Attribute entspricht, z.B. Paketgrößen, Verbindungsdauer, usw. Jedes Objekt aus der Eingabe wird somit durch einen Vektor in diesem Raum dargestellt.

Der k-Means-Algorithmus findet k Cluster, die durch ihre Mittelpunkte im Raum festgelegt werden. Zuerst werden die Mittelpunkte $m_1,...,m_k$ zufällig oder manuell initialisiert. Dann werden wiederholt die folgenden beiden Schritte ausgeführt:

- Zuordnung aller Daten zum nähesten Clustermittelpunkt
- Neuberechnung der Clustermittelpunkte

Beim ersten Schritt kann als Abstandsmaß die euklidische Distanz $d_e$ der Vektoren verwendet werden, die für zwei Punkte $x = (x_1,...,x_n)$ und $y = (y_1,...,y_n)$ folgendermaßen definiert ist:

$$d_e(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Die neuen Clustermittelpunkte m ergeben sich aus den Objekten $x_1,...,x_l$, die zum jeweiligen Schritt dem Cluster zugeordneten sind, zu:

$$m = \frac{1}{l} \sum_{i=1}^{l} x_i$$

[7]

Das Verfahren wird solange fortgesetzt, bis sich keine Veränderung der Mittelpunkte mehr einstellt. Dabei wird die Intra-Cluster-Varianz V minimiert:

$$V = \sum_{i=1}^{k} \sum_{x_j \in C_i} \|x_j - m_i\|^2$$

Wobei k die Anzahl der Cluster $C_i$, i=1...k und $m_i$ der Mittelpunkt des jeweiligen Clusters ist. Der Algorithmus konvergiert immer, allerdings kann es sein, dass er dies nur gegen ein lokales Minimum tut [8].

Die folgenden Bilder veranschaulichen das Verfahren für einen zweidimensionalen Raum und k=3 Clustern:

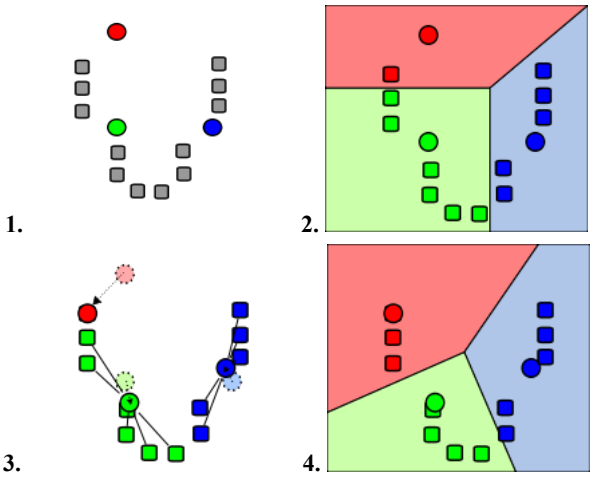


**Abbildung 1. Visualisierung des k-Means-Algorithmus [9].**

Bild1: Die Runden Punkte sind die zufällig gewählten Mittelpunkte der Cluster.
Bild2: Die Datenobjekte werden dem Cluster mit dem nähesten Mittelpunkt zugeordnet.
Bild3: Die Mittelpunkte werden neu berechnet.
Bild4: Neue Einteilung der Cluster.

## 3.2 DBSCAN

DBSCAN betrachtet Cluster als Gebiete mit Objekten die dicht beieinander liegen, die untereinander getrennt von Gebieten geringerer Dichte sind. Im Gegensatz zu k-Means können dadurch die Cluster beliebige Formen annehmen. Bei DBSCAN sind zwei Parameter von Bedeutung: eine gewisse epsilon-Umgebung und eine Mindestanzahl an Objekten. Liegen in der Umgebung eines Objekts mindestens die festgelegte Anzahl an anderen Punkten, wird zu diesem Objekt ein Cluster erstellt. Diesem Cluster werden anschließend alle weiteren Objekte zugewiesen die innerhalb der epsilon-Umgebung der enthaltenen Objekte liegen usw. Die epsilon-Umgebung gibt hierbei die maximale Distanz der Objekte voneinander für eine Zuweisung an. Objekte die am Ende übrig bleiben und keinem Cluster angehören werden als Rauschen gewertet – im Gegensatz zum klassischen k-Means oder AutoClass, bei denen jedes Objekt zugewiesen wird. Als Abstandsmaß wird hier wieder die euklidische Distanz gewählt.

AutoClass arbeitet mit einem Wahrscheinlichkeitsmodell für die Cluster. Der Algorithmus ermittelt von selbst die optimale Anzahl an Clustern und die Objekte können mit unterschiedlichen Wahrscheinlichkeiten mehreren Clustern zugeordnet sein, wobei hier ein Objekt einfach dem wahrscheinlichsten Cluster zugeordnet wird. Jedes Cluster besitzt eine Wahrscheinlichkeitsverteilung, deren Parameter mit einem Expectation Maximization-Algorithmus ermittelt werden (vgl. Kapitel 4.2).

## 3.3 Resultate

Die Testdaten werden mit Hilfe von Payload-Analyse-Tools untersucht, um die tatsächlichen Anwendungen der Verbindungen zu ermitteln. Wurden mehrere Anwendungen in einem Cluster zusammengefasst, werden bei der Klassifizierung alle Verbindungen zur in diesem Cluster dominierenden Anwendung zugeordnet. Die allgemeine Genauigkeit wird wie folgt definiert:

$$overall\ accuracy = \frac{\sum true\ positives\ for\ all\ clusters}{total\ number\ of\ connections}$$

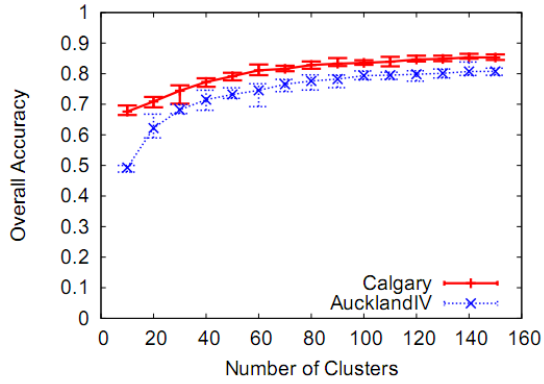


**Abbildung 2. Overall Accuracy bei k-Means [3].**

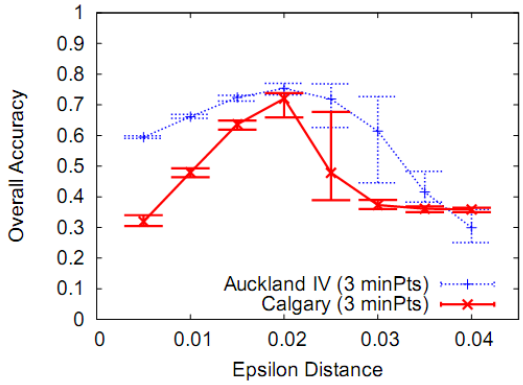


**Abbildung 3. Overall Accuracy bei DBSCAN [3].**

Die Diagramme der Abbildungen 2 und 3 zeigen die ermittelten Ergebnisse mit den verschiedenen Algorithmen. Bei k-Means sieht man die Ergebnisse für verschiedene Anzahlen an Cluster. Die Anzahl muss dem Algorithmus hierbei vorgegeben werden. Man kann hier erwarten das jedem Anwendungstyp mindesten ein Cluster zugewiesen wird. Bei Anwendungstypen mit unterschiedlichem Verhalten wie HTTP (browsing, streaming, …) werden mehrere Cluster entstehen, da jedes Verhalten sich in den untersuchten Attributen unterscheidet.

Auch DBSCAN wurde mit unterschiedlichen Parametern getestet. Hier müssen, wie erwähnt, die Größe der epsilon-Umgebung und die Mindestanzahl minPts an Objekten in einer solchen angegeben werden. Der gewählte Wert von drei für die Mindestanzahl minPts führt zu vielen kleinen Clustern, die ja auch schon bei k-Means zu guten Ergebnissen führten. Für zu große epsilon-Umgebungen verschmelzen die Cluster zu wenigen großen und die Genauigkeit sinkt rapide. Hier sei nochmal die Besonderheit des DBSCAN erwähnt Verbindungen auch als Rauschen klassifizieren zu können. Dadurch wird die allgemeine Genauigkeit zwar gesenkt, denn solche Verbindungen werden als falsch klassifiziert gewertet, jedoch ist das Verhältnis von TP zu FP innerhalb der Cluster bei DBSCAN dadurch am höchsten, so dass es weniger falsch klassifizierte Verbindungen im Verhältnis zu den richtigen gibt. Somit entstehen Cluster mit höherer Präzision was je nach Ziel der Klassifizierung von Vorteil sein kann.

AutoClass erweist sich als der beste Algorithmus in Sachen allgemeine Genauigkeit.

Um später beim Klassifizieren Rechenaufwand zu sparen, was ein wichtiger Faktor bei bei online Klassifikation ist, können kleine Cluster, die nur eine unbedeutende Anzahl an Verbindungen im Vergleich zur Gesamtzahl enthalten, verworfen werden. So verliert man nur wenig Genauigkeit, spart aber Rechenzeit bei der Zuweisung zu den Clustern.

Ein weiterer wichtiger Unterschied zwischen den drei Algorithmen ist, dass das Clustering bei k-Means und DBSCAN in wenigen Minuten, bei AutoClass hingegen erst nach 4,5 Stunden beendet war.

# 4. EXPECTATION MAXIMIZATION

Das hier vorgestellte Verfahren wurde an der Université Pierre et Marie Curie in Frankreich erarbeitet (siehe [1], [2]). Hierbei soll eine Verbindung „on the fly“ schon anhand der ersten paar verschickten Pakete zu einer Anwendung zugeordnet werden. Das Verfahren arbeitet dabei erneut in zwei Phasen und verwendet Clustering (Unsupervised Learning) mit k-Means, Gaussian Mixture Models (GMM) und Hidden Markov Models (HMM). Wobei die letzten beiden stochastische Modelle sind, die mit Expectation Maximization (EM) Verfahren arbeiten.

Folgende Abbildung zeigt einen Überblick des hier vorgestellten Verfahrens:

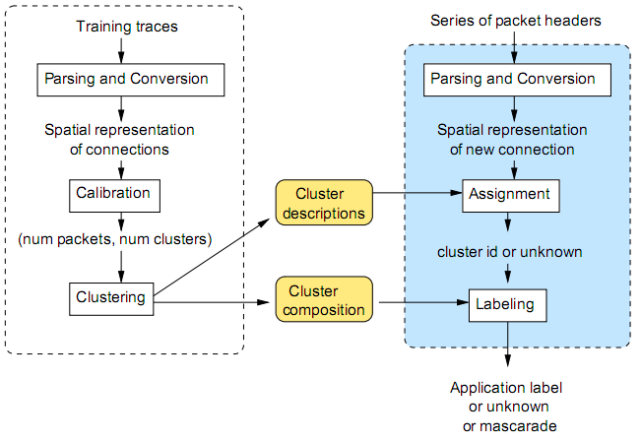


**Abbildung 4. Überblick über das Verfahren [1].**

## 4.1 Offline Training Phase

Als Repräsentation einer TCP-Verbindung werden hier zu jeder Verbindung nur die Größe der ersten paar Pakete und deren Richtung benötigt. Somit kann schon bevor eine Verbindung abgeschlossen ist und ohne großen Aufwand eine Zuordnung erfolgen. Im Schritt „Parsing and Conversion“ werden aus dem verwendeten Trace die benötigten Daten extrahiert.

Im Schritt „Calibration“ wird die optimale Anzahl der verwendeten Cluster und die Anzahl der zu untersuchenden Pakete ermittelt. Die optimale Anzahl der Cluster wird wieder ermittelt indem man die Ergebnisse des Clustering-Algorithmus mit der perfekten Zuordnung vergleicht (Payload-Analyse). Hier wurden 40 Cluster als guter Wert ermittelt. Die ersten fünf Pakete nach dem Three-Way-Handshake erweisen sich schon als aussagekräftig, da hier die Verbindungsverhandlung auf Anwendungsebene stattfindet.

## 4.2 Gaussian Mixture Models

Bei GMMs handelt es sich um einen Expectation Maximization-Algorithmus. EM-Algorithmen liefern keine harte Zuordnung der Daten zu den Clustern, sondern es wird für jedes Objekt die Wahrscheinlichkeit der Zugehörigkeit zu jedem der Cluster ermittelt. Dabei wird davon ausgegangen, dass die Art der Wahrscheinlichkeitsverteilung der Daten bekannt ist. Oft wird hier die Normalverteilung angenommen. Dabei wird meist eine Mischverteilung, d.h. eine gewichtete Summe von so vielen Verteilungen wie Cluster vorhanden sind, verwendet. Ein EM-Algorithmus bestimmt nun die Parameter (Mittelwert und Standardabweichung) der Verteilung für jeden Cluster.

Ähnlich wie bei k-Means werden wiederholt die beiden folgenden Schritte ausgeführt:

- **Expectation**: Für jedes Objekt x wird berechnet, mit welcher Wahrscheinlichkeit $P(C_j|x_i)$ es zu jedem Cluster $C_j$ gehört.
- **Maximization**: Unter Verwendung der neu berechneten Wahrscheinlichkeiten werden die Parameter der Verteilung mit Maximimum-Likelihood-Methoden neu berechnet.

Erneut wird dies so lange fortgesetzt bis sich keine Änderung der Parameter mehr einstellt, was immer eintreten wird, jedoch unter Umständen nur bei einem lokalen Optimum [7, 8].

## 4.3 Hidden Markov Models

Das HMM ist ein stochastisches Modell, dem eine Markovkette zugrunde liegt. Die Zustände der Markovkette sind dabei „unsichtbar". Bei jedem Zeitschritt wird aber vom aktuellen Zustand ein sichtbares Symbol erzeugt. Beim Training von HMM werden Expectation Maximization-Algorithmen verwendet, die die Parameter, also Übergangswahrscheinlichkeiten zwischen den Zuständen und die jeweiligen Beobachtungswahrscheinlichkeiten für die Symbole, des HMM ermitteln.

Man kann dann bei gegebener Symbolfolge das HMM ermitteln, welches am wahrscheinlichsten die Folge erzeugt hat. Im hier vorliegenden Fall bestehen die Symbolfolgen aus den Paketgrößen der Verbindungen und jeder Cluster wird durch ein HMM repräsentiert, welches über so viele Zustände wie die Anzahl der betrachteten Pakete verfügt.

Schlussendlich werden die Cluster noch mit den darin enthaltenen Anwendungen gekennzeichnet.

## 4.4 Online Classification Phase

Beim „Assignment" werden nun „on the fly" Paket-Header aufgezeichnet und sobald drei Pakete einer Verbindung vorhanden sind, kann diese schon einem Cluster zugewiesen werden. Die Verfahren dazu ergeben sich aus dem verwendeten Clustering-Algorithmus. Für k-Means kann die Verbindung dem am nähesten liegenden Cluster zugeordnet werden, für GMM und HMM werden Maximum Likelihood-Methoden verwendet.

Beim „Labeling" wird die Verbindung schließlich mit der für sie ermittelten Anwendung gekennzeichnet. Hierbei wird einfach die im Cluster vorherrschende Anwendung verwendet, die in der Trainingsphase ermittelt wurde. Dabei werden leider alle Verbindungen die eigentlich zu einer anderen Anwendung in diesem Cluster gehören falsch klassifiziert. Zusätzlich wurde außerdem noch die Verwendung der Portnummer untersucht, um eine Verbindung möglicherweise doch einer anderen Anwendung im Cluster zuzuweisen. Allerdings nur wenn dies sinnvoll ist, also z.B. wenn es sich um einen der Well-Known-Ports handelt. Eine weitere Einsatzmöglichkeit für die Betrachtung der Portnummer ist das Erkennen von Täuschungsversuchen. Sollte eine Verbindung einen Well-Known-Port verwenden und einem Cluster zugeordnet werden, der die eigentlich zum Port gehörende Anwendung nicht enthält, dann kann dies ein Hinweis auf eine Anwendung sein, die versucht als sicher eingestufte Ports zu verwenden, um Firewalls zu umgehen. Solche Verbindungen könnten speziell gekennzeichnet werden und bei häufigem Auftreten könnte ein Netzwerk-Administrator dem Verdacht nachgehen.

Verwendet man Algorithmen, die Wahrscheinlichkeiten für die Zuordnung der Verbindungen verwenden, so kann zusätzlich ein Grenzwert für diese angegeben werden. Sollte eine Verbindung für alle Cluster unter dieser Grenzwahrscheinlichkeit liegen, wird ihre Anwendung als unbekannt eingetragen. Steigende Anzahlen von unbekannten Einträgen könnten dann auf eine neue Anwendung im Netzwerk hinweisen. Dann wäre ein erneutes Training zur Erkennung der neuen Anwendung angebracht.

## 4.5 Resultate

Tabelle 2 zeigt, nach den verschiedenen Anwendungen aufgeschlüsselt, die Ergebnisse von GMM und HMM beim Klassifizieren eines Test-Trace. Die Prozentangaben beziehen sich jeweils auf die Anzahl der Verbindungen. Die Tabelle zeigt die Ergebnisse, wenn beim Labeling die im Cluster vorherrschende Anwendung verwendet wird. Außerdem wurden keine Grenzwerte für die Wahrscheinlichkeit verwendet – es wird also nichts als unbekannt eingestuft. Für k-Means existiert im Paper [1] keine Aufschlüsselung nach Anwendungen, es lieferte aber auch eine Gesamt-Genauigkeit von über 90%. Der 0,0% Wert von HMM bei POP3 entsteht dadurch, dass POP3 und NNTP im selben Cluster sind und NNTP dort vorherrschend ist. Tabelle 1 zeigt die Erkennung von unbekannten Anwendungen, die in den Trainingsdaten nicht vorhanden waren, durch GMM bei einem Grenzwert von 99% - d.h. eine Verbindung wird als unbekannt eingestuft, wenn für keines der Cluster die Zugehörigkeits-wahrscheinlichkeit über 99% liegt.

**Tabelle 1. Erkennung unbekannter Anwendungen [1].**

| | GMM mit Grenzwert 99% | |
|---|---|---|
| Anwendung | FP | Unbekannt |
| Bittorent | 33,2% | 66,8% |
| Gnutella | 100,0% | 0,0% |
| IRC | 10,0% | 90,0% |
| LDAP | 8,8% | 91,2% |
| MSN | 38,5% | 61,5% |
| Mysql | 0,0% | 100,0% |

**Tabelle 2. Labeling Accuracy von GMM und HMM [1].**

| | GMM | | HMM | |
|---|---|---|---|---|
| Anw. | TP | FP | TP | FP |
| NNTP | 81,2% | 0,0% | 99,8% | 3,6% |
| POP3 | 96,5% | 0,7% | 0,0% | 0,0% |
| SMTP | 90,1% | 0,1% | 83,6% | 0,2% |
| SSH | 89,4% | 0,0% | 89,6% | 0,0% |
| HTTPS | 60,1% | 0,0% | 53,1% | 0,0% |
| POP3S | 93,4% | 1,1% | 96,1% | 1,1% |
| HTTP | 96,2% | 1,3% | 99,0% | 1,7% |
| FTP | 92,4% | 0,4% | 92,9% | 0,3% |
| Edonkey | 94,1% | 0,3% | 71,4% | 0,0% |
| Kazaa | 88,9% | 2,6% | 67,7% | 0,7% |
| Overall | 93,7% | X | 92,3% | X |

# 5. CLASSIFICATION

An der University of Adeleide in Australien wurde die Möglichkeit untersucht mit Supervised Learning-Methoden den Netzwerkverkehr in Verhaltensklassen einzuteilen, für die unterschiedliche Quality-of-Service (QoS) Anforderungen gelten sollen [5]. Verschiedene Anwendungstypen besitzen unterschiedliche solche Anforderungen an die Netzwerkverbindung, wie z.B. hoher Datendurchsatz für Datentransfer und kurze Verzögerungszeiten für interaktive Programme. Um diesen Anforderungen gerecht zu werden, sind Netzwerkbetreiber, besonders von großen Unternehmen, deshalb an der Möglichkeit interessiert ihren Datenverkehr verschiedenen QoS-Klassen zuordnen zu können. So könnten Anwendungen, die

für das Unternehmen wichtig sind, priorisiert werden und allgemein das vorhandene Netz effektiver ausgenützt werden.

Die Festlegung des QoS für eine Verbindung könnte im Prinzip auch von den Endpunkten aus durch die jeweilige Anwendung erfolgen. Aus Vertrauensgründen und der Skalierbarkeit der nötigen Administration ist es aber von Vorteil diese im Netzwerk selbst durchzuführen.

In dem hier verwendeten Verfahren wird nicht mehr versucht jeder Verbindung konkret eine Anwendung zuzuordnen, sondern diese werden nach ihrem Verhalten unterschieden. Hierzu wurden vier Verhaltensklassen definiert:

- Interactive: Beinhaltet Anwendungen bei denen ein Benutzer in Echtzeit mit einem entfernten System interagiert. Darunter fallen z.B. Remote-Login-Sessions oder ein interaktives Web-Interface.
- Bulk Data Transfer: Für die Übertragung von großen Datenmengen ohne Echtzeit-Anforderungen.
- Streaming: Multimedia Anwendungen die in Echtzeit übertragen, z.B. Videos.
- Transactional: Beinhaltet Datenverkehr, der eine kleine Anzahl von Anfrage/Antwort-Paaren übermittelt, die jeweils zu einer Transaktion zusammengefasst werden können. Darunter fallen z.B. DNS oder Oracle Transaktionen.

Zum Training werden hier wieder Datensätze, die als Referenz für die einzelnen Klassen verwendet werden können benötigt. In diesem Fall wurden für jede Klasse ein/zwei gut bekannte Anwendungen verwendet, die möglichst ausschließlich die jeweiligen Verhaltensweisen zeigen und außerdem weit verbreitet sind, um einen repräsentativen Datensatz zu erhalten.

Gewählt wurden jeweils:

- Interactive: Telnet
- Bulk Data Transfer: FTP-Data, Kazaa
- Streaming: RealMedia Streaming
- Transactional: DNS, HTTPS

Da die Daten hier schon mit den Klassenzugehörigkeiten versehen sind, handelt es sich um Supervised Learning. Klassifiziert werden Tupel aus Server-IP und Server-Port. Zu jedem solchen Tupel wird eine Statistik über verschiedene Eigenschaften der Verbindungen zum Server gesammelt. Wurden genügend Werte in der Statistik gesammelt, kann die Klassifikation des Tupels beginnen. Anschließend können alle Pakete mit diesem Ziel-Port und Ziel-IP automatisch in die entsprechende Class-of-Service zugeordnet werden.

Zur Klassifizierung wurden zwei einfache, aber verbreitete Methoden verwendet: Nearest Neighbour (NN) und Linear Discriminant Analysis (LDA).

## 5.1 Nearest Neighbour

Bei NN wird ein neues Objekt einfach der Klasse des zu ihm nähesten (euklidischer Abstand) Objekts aus den Trainingsdaten zugeordnet. Diese Technik lässt sich noch zu k-NN erweitern, so dass die k-nähesten Objekte ermittelt werden und unter diesen die am stärksten vertreten Klasse gewählt wird.

## 5.2 Linear Discriminant Analysis

Bei LDA werden die Klassen wieder mit Wahrscheinlichkeitsverteilungen beschrieben. Dabei wird die Klasse zur Zuordnung gewählt, die bei gegebenem Eingabeobjekt am wahrscheinlichsten ist. LDA liefert die Funktion die diese Wahrscheinlichkeit berechnet, wobei die Varianz zwischen den Klassen im Verhältnis zu der Varianz innerhalb der Klassen maximiert wird, so dass diese sich minimal überlappen.

## 5.3 Resultate

Abbildungen 5 und 6 zeigen die Einteilung von zwei Testdatensätzen in die vier Cluster, wobei als Attribute zur Zuordnung die durchschnittliche Paketgröße und die durchschnittliche Dauer der TCP-Verbindungen gewählt wurden. Zur Erinnerung: ein Punkt entspricht hier einem Tupel aus Server-IP und -Port.

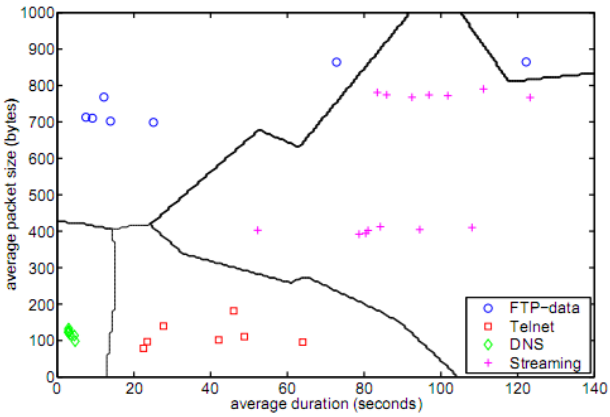


**Abbildung 5. Nearest Neighbour [5].**

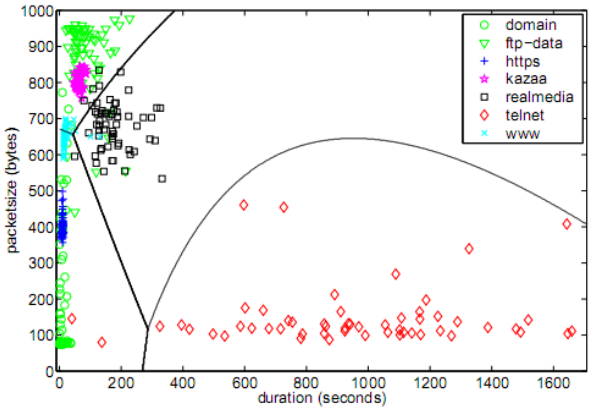


**Abbildung 6. LDA [5].**

Wie man in Abbildung 6 sehen kann, sind Streaming und Bulk Data Transfer am schwierigsten zu unterscheiden. Deswegen wurden weitere Attribute untersucht mit denen man diese genauer unterscheiden kann. Die Resultate davon kann man in Abbildung 7 erkennen.

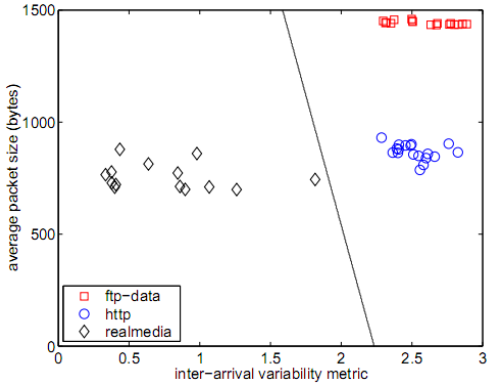


**Abbildung 7. Streaming (RealMedia) vs. Bulk Data (FTP) [5].**

Hier wurden nun die durchschnittliche Paketgröße und die Inter-Arrival-Variablity verwendet. Letztere ist das Verhältnis vom Mittel der Inter-Arrival-Times und der zugehörigen Standardabweichung, zusammen wiederum gemittelt über alle Verbindungen eines Tages. Diese eignet sich zur Unterscheidung, da die Zeitabstände zwischen den Paketen beim Streaming regelmäßiger sind als beim Bulk Data-Verkehr.

Zum Schluss wurden zur weiteren Überprüfung der Ergebnisse noch zwei Traces ein und desselben Netzes, aber zu unterschiedlichen Zeiten, verwendet. Der erste Trace wurde zum Training benützt, und der zweite, sechs Monate ältere, wurde anschließend klassifiziert. Während die Server-Tupel der Anwendungen zwar nicht mehr an der exakt gleichen Stelle waren, lagen sie immer noch in den ermittelten Grenzen der Cluster.

# 6. FAZIT

Verschiedene Forschungsgruppen beschäftigen sich mit dem Einsatz von maschinellem Lernen um alte Verkehrs-klassifizierungsmethoden zu verbessern. Der neue Ansatz liefert dabei erfolgversprechende Ergebnisse. Dabei ergeben sich sogar neue Möglichkeiten wie das automatische Erkennen von neuen Anwendungen und das Feststellen von verdächtigem Verhalten im Netz. Eine weitere etwas andere Anwendungsmöglichkeit, die hier nicht vorgestellt wurde ist z.B. der Versuch Kommunikation von Botnetzen zu erkennen [10].

Wie sich gezeigt hat gibt es unzählige Kombinationsmöglichkeiten von verschiedenen Techniken, Algorithmen und verwendeten Attributen des Netzwerkverkehrs. Es gibt deswegen noch Forschungsbedarf, welche davon sich als am zuverlässigsten erweisen und welche weiteren Anwendungs-möglichkeiten Verfahren dieser Art bieten.

Außerdem besteht auch weiterhin die Möglichkeit einer korrekten Erkennung durch die neuen Techniken zu entgehen, indem die untersuchten Kriterien durch ein Anwendungs-programm zufällig oder gezielt variiert werden. Wie sich die Verfahren unter solchen Bedingungen schlagen muss sich auch noch zeigen [1].

# 7. Literatur


[1] Laurent Bernaille, Renata Teixeira, and Kavé Salamatian. Early Application Identification. Université Pierre et Marie Curie - LIP6, CNRS, Paris, France. http://www-rp.lip6.fr/site_npa/site_rp/_publications/737-conextFinal.pdf.

[2] Laurent Bernaille, Renata Teixeira, Ismael Akodjenou, Augustin Soule, and Kave Salamatian. Traffic Classification On The Fly. Université Pierre et Marie Curie - Paris VI, Thomson Paris Lab. http://www-rp.lip6.fr/site_npa/site_rp/_publications/714-ccredito.pdf.

[3] Jeffrey Erman, Martin Arlitt, and Anirban Mahanti. Traffic Classification Using Clustering Algorithms. University of Calgary, 2500 University Drive NW, Calgary, AB, Canada. http://conferences.sigcomm.org/sigcomm/2006/papers/minenet-01.pdf.

[4] Anthony McGregor, Mark Hall, Perry Lorier, and James Brunskill. Flow Clustering Using Machine Learning Techniques. PAM(Passive&Active Measurement Workshop)2004. http://www.pam2004.org/papers/166.pdf.

[5] Matthew Roughanx, Subhabrata Sen, Oliver Spatscheck, and Nick Duffield. Class-of-Service Mapping for QoS: A Statistical Signature-based Approach to IP Traffic Classification. School of Mathematical Sciences, University of Adelaide,SA 5005, Australiax, AT&T Labs – Research, Florham Park, NJ 07932-0971, USA. http://www.imconf.net/imc-2004/papers/p135-roughan.pdf.

[6] Christian Osendorfer, and Martin Felder. Skript der Vorlesung "Machine Learning I"(2008). Technische Universität München. http://www6.in.tum.de/pub/Main/TeachingWs2008MachineLearning/Slides_01_Bayes.pdf.

[7] Wolfgang Ertel. Grundkurs Künstliche Intelligenz – Eine praxisorientierte Einführung.vieweg, 1. Auflage 2008, S.227-229(k-Means, Expectation Maximization) und S.184(Data Mining).

[8] Christopher M. Bishop. Pattern Recognition and Machine Learning. Springer Science + Business Media, LLC, 9. Auflage 2007, S.424 – 427(k-Means) und S. 430ff.(GMM).

[9] Englische Wikipedia. http://en.wikipedia.org/wiki/K-means.

[10] Carl Livadas, Bob Walsh, David Lapsley, Tim Strayer. Using Machine Learning Techniques to Identify Botnet Traffic. http://www.ir.bbn.com/documents/articles/lcn-wns-06.pdf.