max planck institut
informatik

Ideen der Informatik

Maschinelles Lernen

Kurt Mehlhorn
Adrian Neumann
Max-Planck-Institut für Informatik

Übersicht

- Lernen: Begriff
- Beispiele für den Stand der Kunst
- Spamerkennung
- Handschriftenerkennung
  - mit und ohne Trainingsdaten
- Neuronale Netzwerke
  - Maschinelles Sehen

Lernen

- Fähigkeit, Verhalten zu verbessern aufgrund von Erfahrungen
- Verallgemeinern von Erfahrungen
- Informatik: Programmieren durch Beispiele anstatt durch Angabe eines Programms
- Ein Lernalgorithmus entwickelt das Programm aus (vielen) Daten

Typische Anwendungen

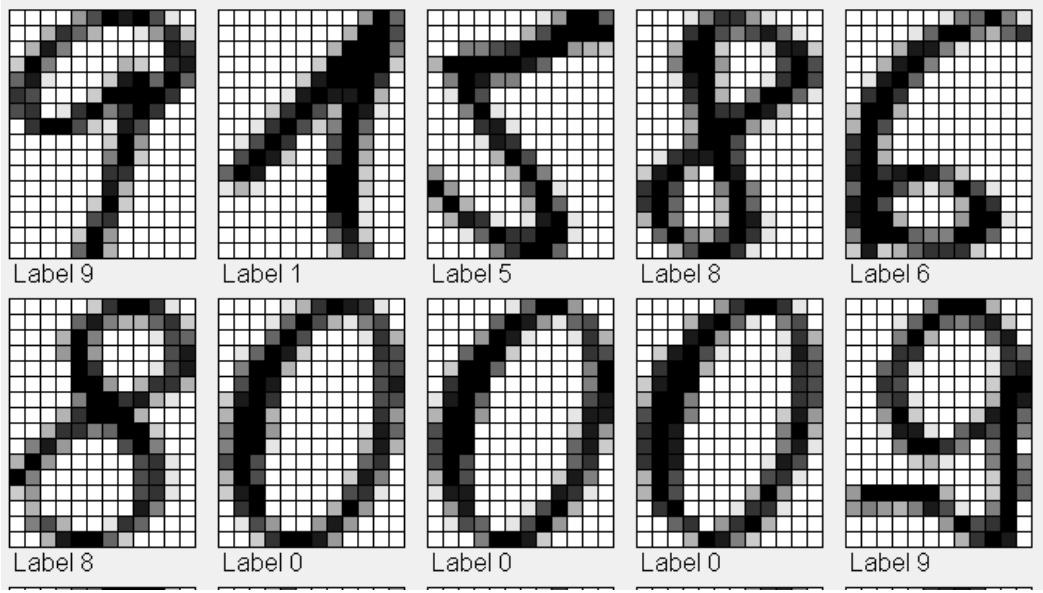
- Klassifikation: Spam versus Ham, Ziffernerkennung, Verkehrszeichenerkennung, Objekte auf Bildern, Identifikation von Personen auf Bildern, Handlungen aus Videosequenzen
- Robotersteuerung, lerne Autofahren
- Spracherkennung, Sprachsteuerung

Potential

- Suchmaschinen mit Bild / Sprach-Anfragen
- Personenerkennung auf Videos
- XXX mit gesprochener Sprache
- Selbstfahrende Autos
- Bessere Benutzerschnittstellen
- Maschinelle Übersetzung

Arten von Lernen

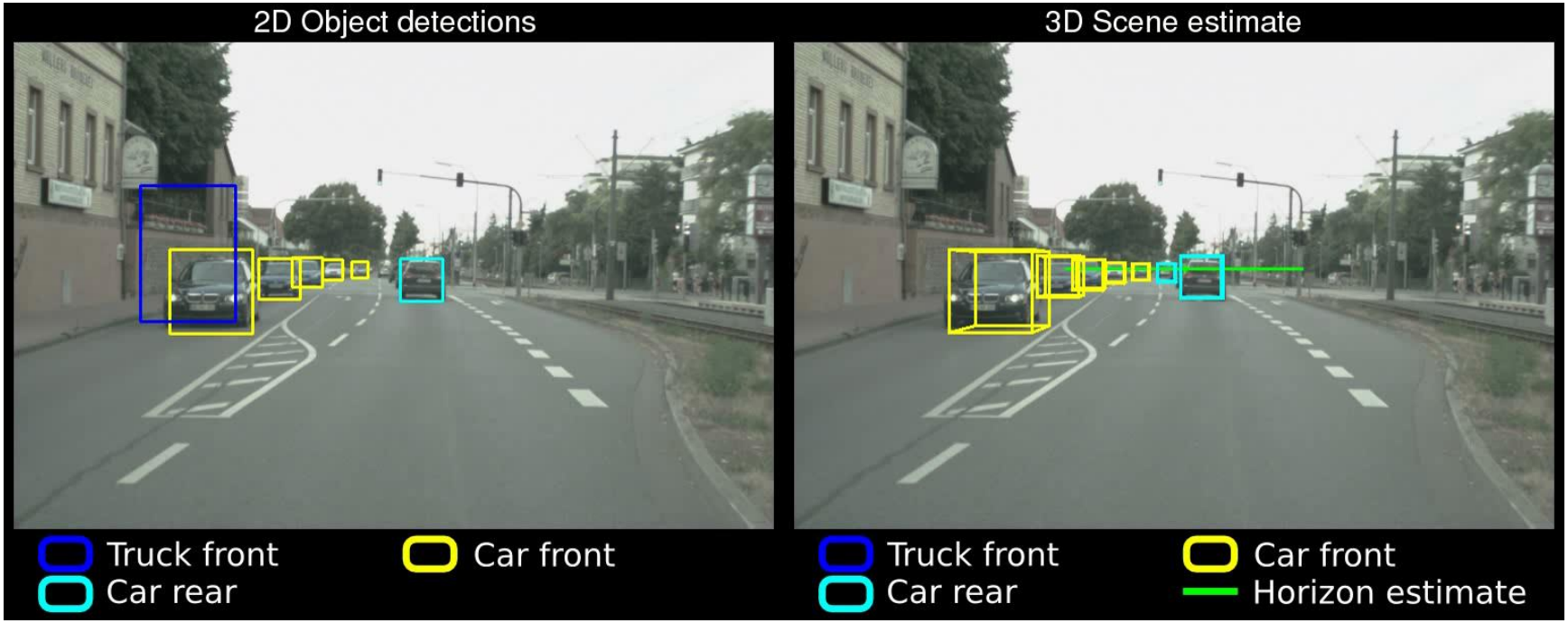
- **Supervised**: mit Trainingsdaten oder sogar mit Lehrer



- **Unsupervised**: ohne Trainingsdaten; dann ist es mehr Entdecken als Lernen

Objekterkennung

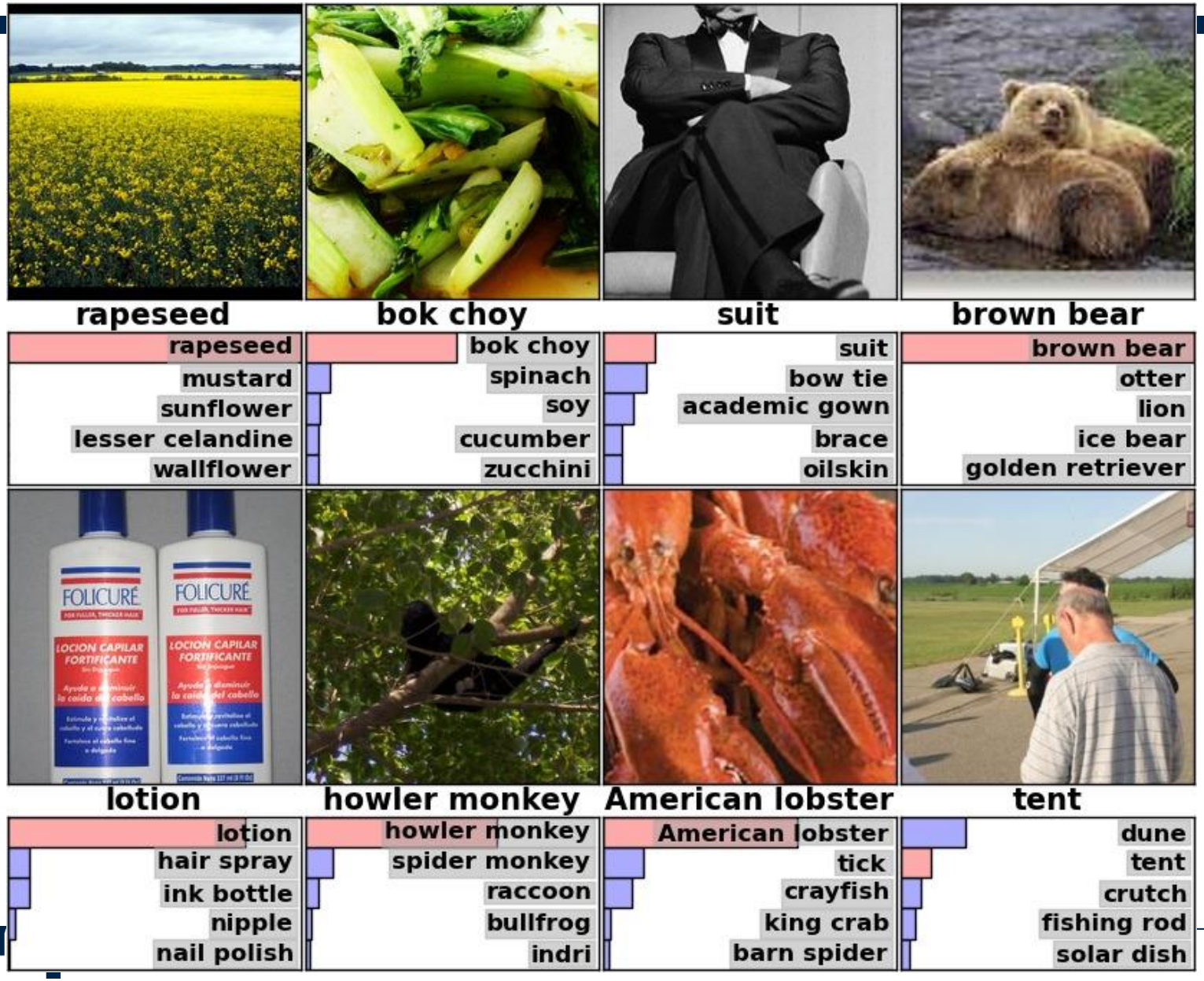
Abteilung Schiele: MPI für Informatik



Personenerkennung

Abteilung Schiele: MPI für Informatik

Klassifikation (Krizhevsky et al, 2012)



Suche (Krizhevsky et al, 2012)

Spamerkennung

Spam = unerwünschte Nachrichten

Ham = erwünschte Nachrichten

Wir lernen einen Bayes‘schen Filter kennen

Bayessche Regel

(englischer Pfarrer und Mathematiker, 1701 – 1761)

90% der Früchte in einem Sack sind Äpfel und 10% sind Paprika. Von den Äpfeln sind 10% rot und 90% grün. Bei den Paprika sind es jeweils 50%.

Ich entnehme eine Frucht zufällig. Sie ist rot. Was für eine Frucht ist es?

- Bayes: entscheide dich für den wahrscheinlicheren Fall und den bestimmt man so.

Aufgabe

- 5% der Bevölkerung erkranken an der Krankheit *X*. Ein Test führt bei gesunden Patienten zu 30% zu einem positiven Ergebnis, bei kranken Patienten zu 100%. Wie hoch ist die Wahrscheinlichkeit bei einem positiven Testergebnis tatsächlich krank zu sein?

Bayes'sche Regel

90% der Früchte sind Äpfel, 10% sind Paprika. Von den Äpfeln sind 10% rot und 90% grün. Bei den Paprika sind es jeweils 50%.

$$P(\text{Apfel} \mid \text{rot}) = \frac{\#\text{ rote Äpfel}}{\#\text{ rote Früchte}} =$$

(Prozentsatz der Äpfel unter den roten Früchten)

Spam versus Ham (Junk Mail)

- Absenderbasiert
  - E-Mail von Bekannten ist kein Spam
  - Schwarze Listen

- Inhaltsbasiert
  - Nutzer klassifiziert E-Mails als gut und schlecht; System lernt daraus; Nutzer muss immer weniger eingreifen

Inhaltsbasierte Filter

- In der Trainingsphase lernen wir
  - Wahrscheinlichkeit von Ham und Spam
  - Jeweils Wahrscheinlichkeiten für Worte

- 70% ist Ham, 30% ist Spam

- Ham

Freund	Vorle-sung	Algorith-mus	Geld	Viagra	schnell
0.1	0.3	0.3	0.1	0.1	0.1

- Spam

Freund	Vorle-sung	Algorith-mus	Geld	Viagra	schnell
0.2	0.1	0.1	0.2	0.3	0.1

Trainingsphase

- Nutzer klassifiziert E-Mails als Spam und Ham (damit beide Wahrscheinlichkeiten)

- Sei *n (m)* die Gesamtlänge meiner guten (schlechten) E-Mails (in Worten), sei *v (w)* die Anzahl der Vorkommen eines bestimmten Wortes

- Wahrscheinlichkeit des Wortes in Ham = $\frac{v}{n}$

- Wahrscheinlichkeit des Wortes in Spam = $\frac{w}{m}$

Inhaltsbasierte Filter

- Ham

Freund	Vorle-sung	Algorith-mus	Geld	Viagra	schnell
0.1	0.3	0.3	0.1	0.1	0.1

- Spam

Freund	Vorle-sung	Algorith-mus	Geld	Viagra	schnell
0.2	0.1	0.1	0.2	0.3	0.1

- Viagra Geld Freund

P(Text | Ham) = 0.1 · 0.1 · 0.1 = 1/1000

P(Text | Spam) =

- Bei 70% Ham und 30% Spam

P(Ham | Text) =

Inhaltsbasierte Filter

- Ham

Freund	Vorle-sung	Algorith-mus	Geld	Viagra	schnell
0.1	0.3	0.3	0.1	0.1	0.1

- Spam

Freund	Vorle-sung	Algorith-mus	Geld	Viagra	schnell
0.2	0.1	0.1	0.2	0.3	0.1

- Vorlesung Algorithmus schnell

Falls Ham

Falls Spam

- Bei 70% Ham und 30% Spam

P(Ham | Text) =

Inhaltsbasierte Filter

- Ham

Freund	Vorle-sung	Algorith-mus	Geld	Viagra	schnell
0.1	0.3	0.3	0.1	0.1	0.1

- Spam

Freund	Vorle-sung	Algorith-mus	Geld	Viagra	schnell
0.2	0.1	0.1	0.2	0.3	0.1

- Viagra Algorithmus Vorlesung
- Bei 10% Ham und 90% Spam

Falls Ham: $0.1 \cdot 0.3 \cdot 0.3 = \frac{9}{1000}$

Falls Spam: $0.3 \cdot 0.1 \cdot 0.1 = \frac{3}{1000}$

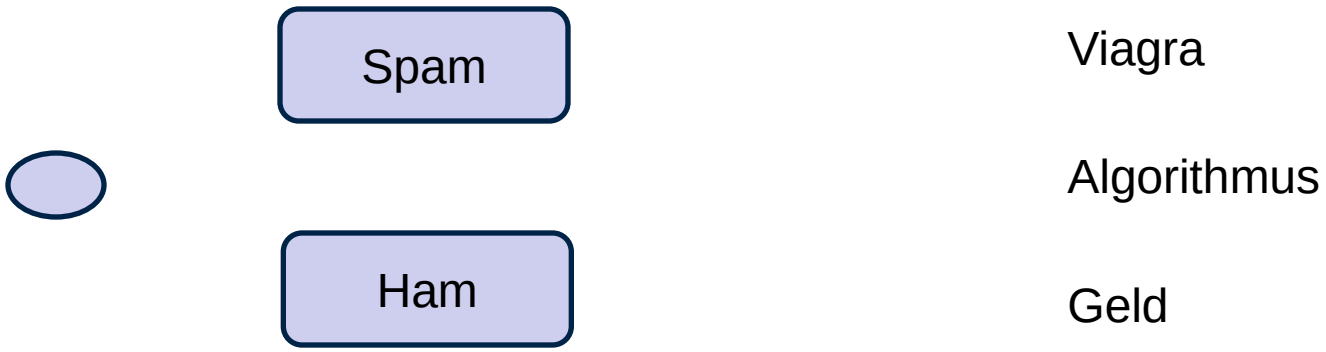
P(Ham | Text) =

Nutzungsphase

- Nutzungsphase: System klassifiziert
- Verteilung wird weiter trainiert (seltene Worte)
- Nutzer kann widersprechen
- Spammer lernen auch dazu: V!agra statt Viagra

Zusammenfassung

- Wir haben Modell, wie Ereignisse (E-Mails) erzeugt werden



- Lernen das Modell in der Trainingsphase
- Geben für jedes Ereignis die wahrscheinlichste Erklärung (Bayes)
- Klassifizierung in: Geschäftspost, Privatpost, Spam

Ziffernerkennung Übersicht

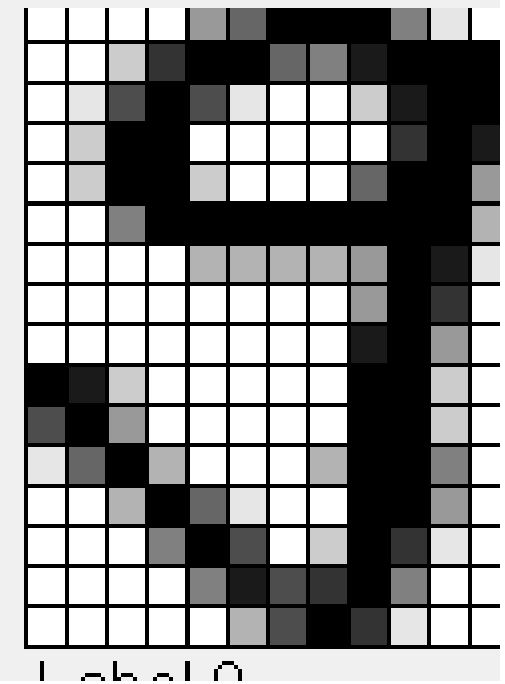
- Darstellung von Bildern in Rechnern
- Trainingsdaten: handgeschriebene Ziffern
- Supervised Learning: mit Label (die Ziffer)
- Unsupervised Learning: ohne Label

Bilder = Matrizen von Zahlen

Ziffer = 12 x 16 Matrix von Grauwerten in [0,1]

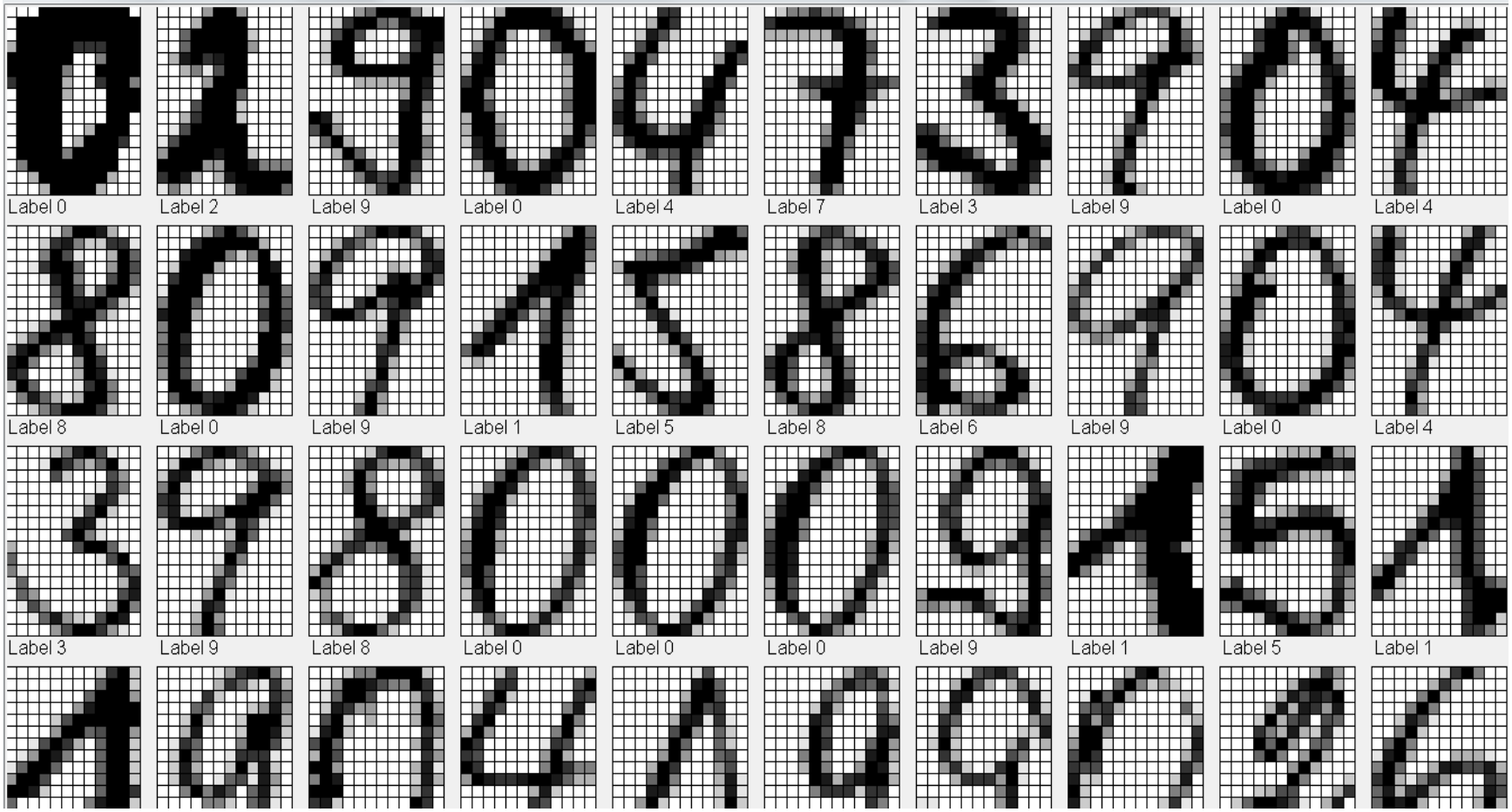
Vektor von Grauwerten der Länge 192

0.0	0.0	0.0	0.2	0.3	0.4	0.8	1.0	1.0	0.7	0.3	0.1
0.0	0.5	0.8	0.9	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.5
0.3	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.9
0.3	1.0	1.0	1.0	1.0	1.0	0.8	1.0	1.0	1.0	1.0	0.8
0.3	1.0	1.0	1.0	1.0	0.8	0.1	0.9	1.0	1.0	0.8	0.2
0.0	0.7	1.0	1.0	1.0	0.8	0.8	1.0	1.0	1.0	0.4	0.0
0.0	0.2	1.0	1.0	1.0	1.0	1.0	1.0	0.9	0.3	0.0	0.0
0.0	0.1	0.7	1.0	1.0	1.0	1.0	0.8	0.1	0.0	0.0	0.0
0.0	0.6	1.0	1.0	1.0	1.0	0.9	0.1	0.0	0.0	0.0	0.0
0.6	1.0	1.0	1.0	1.0	1.0	1.0	0.3	0.0	0.0	0.0	0.0
0.8	1.0	1.0	0.5	0.1	0.7	1.0	0.8	0.2	0.0	0.0	0.0
0.5	1.0	1.0	0.3	0.0	0.0	0.9	1.0	0.9	0.1	0.0	0.0
0.4	1.0	1.0	0.3	0.0	0.0	0.5	1.0	1.0	0.5	0.0	0.0
0.0	0.4	1.0	1.0	0.5	0.3	0.5	1.0	1.0	1.0	0.2	0.0
0.0	0.0	0.5	1.0	1.0	1.0	1.0	1.0	1.0	0.8	0.1	0.0
0.0	0.0	0.0	0.2	0.5	0.7	1.0	1.0	0.9	0.3	0.0	0.0



Ihr Gehirn sieht Ziffern, Ihr Auge und Computer sehen nur eine Matrix von Grauwerten

Trainingsdaten



Ziemlich gutmütig

Grundidee

- Zwei Bilder repräsentieren die gleiche Ziffer, wenn die Bilder sich ähnlich sind.

- Ähnlich = ähnliche Grauwertverteilung

Ähnlichkeit von Vektoren

- Zwei Vektoren *x* und *y* sind ähnlich,
  - wenn *x* – *y* kurz ist
  - wenn der aufgespannte Winkel klein ist

- Länge eines Vektors $x = (x_1, \ldots, x_n)$

$$|| x || = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}$$

- Winkel zwischen *x* und *y*

$$\cos \alpha = \frac{x \cdot y}{||x|| \cdot ||y||}$$

Verfahren: Nearest Neighbor

Um die Bedeutung des Bildes *p* zu finden, finde das Trainingsbild *x* mit *dist(p,x)* minimal (durch lineare Suche über alle Trainingsdaten)

Gib das Label von *x* aus

- Erkennungsrate mit Euklidschem Abstand 0.934
- Mit cos-Distanz 0.940

Detaillierte Ergebnisse

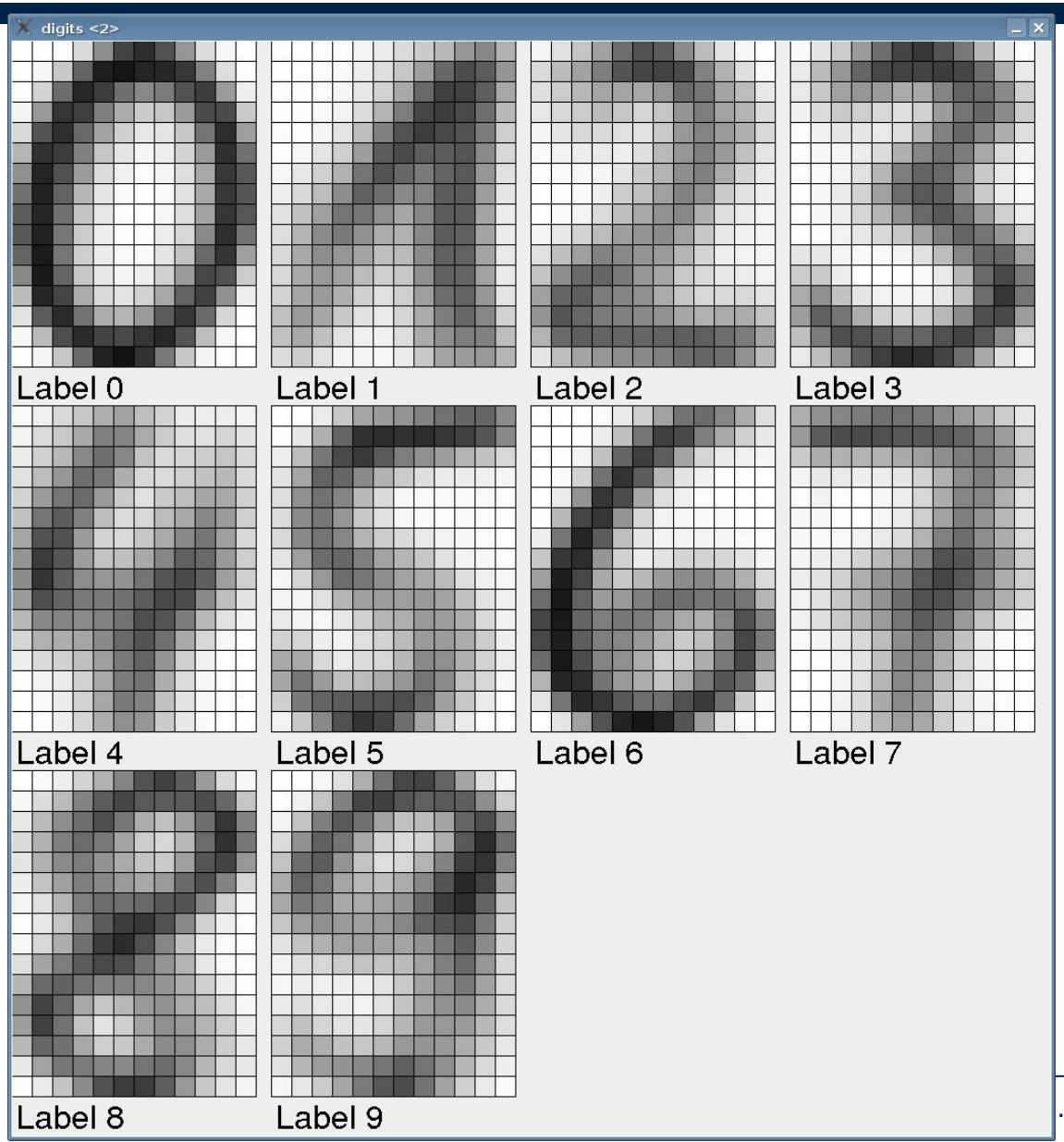
digit = 0 accuracy = 1.0
digit = 1 accuracy = 0.90
digit = 2 accuracy = 0.92
digit = 3 accuracy = 1.0
digit = 4 accuracy = 0.95
digit = 5 accuracy = 0.85
digit = 6 accuracy = 0.84
digit = 7 accuracy = 1.0
digit = 8 accuracy = 0.7
digit = 9 accuracy = 0.94

Klassifizierung ist recht gut, aber sie dauert sehr lang, da jedes Mal ALLE Trainingsdaten angeschaut werden

Klassen → Klassenzentren

- Vorbereitung: Berechne für jede Klasse (Ziffer) das Klassenzentrum durch Durchschnittsbildung.
  (siehe nächste Folie)
- Suche: finde das nächstgelegene Zentrum (10 Vergleiche)
- Erkennungsrate: 0.854
- Mit cos-distance 0.894
- Sehr effizient, aber schlechter

Die Klassenzentren



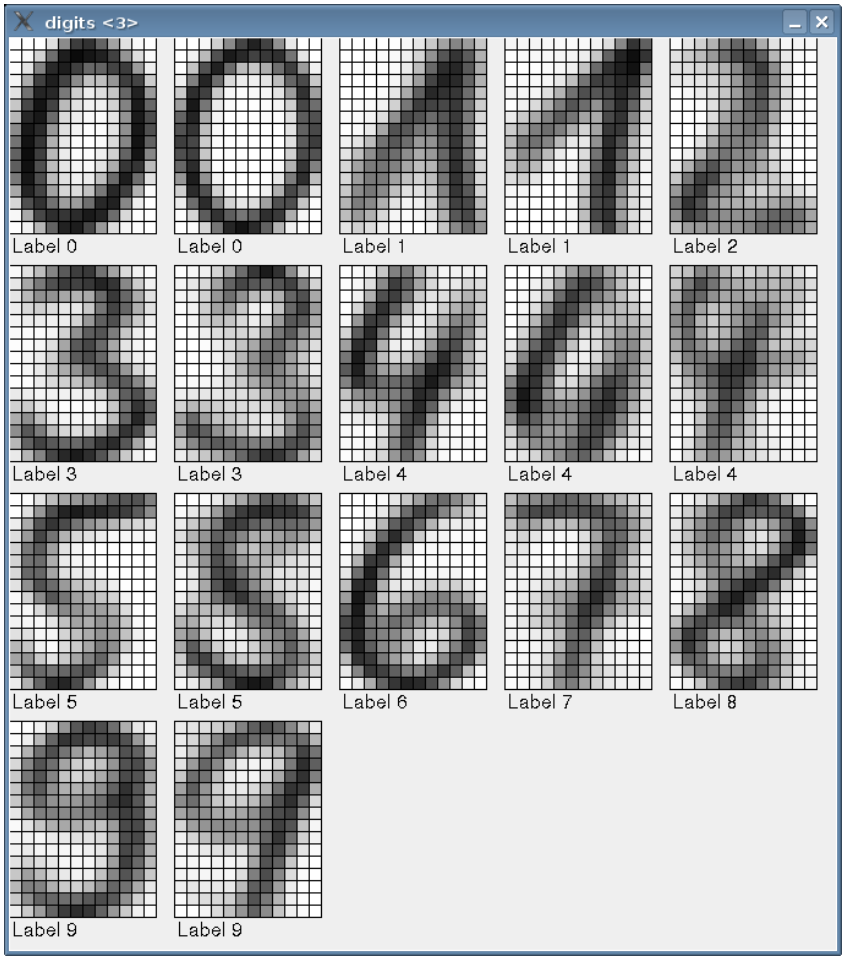
Frage

- Warum genau 10 Klassen?

- Es gibt doch deutlich verschiedene Schreibweisen der Eins, der Neun, der Vier, ….

- Wäre es nicht toll, wenn wir die Klassen automatisch bestimmen könnten?

k-Means Algorithmus



- Automatische Klassifizierung in 17 Klassen
- Danach (!!!) Zuweisung eines Labels per Hand
  und Wegwerfen von schlechten Zentren
- Identifiziert die zwei Schreibweisen der Neun

Unsupervised Lernen

- Vorbereiten der Trainingsdaten ist mühsam
- Können wir Klassen entdecken, ohne dass uns Klassenlabels gesagt werden?
- Automatische Klassifizierung durch *k*-Means Algorithmus

- Danach Vergleich mit den Klassenzentren
- *k* = 10, Rate 0.683
- *k* = 17, 0.733

- with cos-distance,
- *k* = 10, 0.728
- *k* = 17, 0.783
- *k* = 30, 0.864

k-Means Algorithmus

Teilt *n* Punkte in *k* Cluster (Haufen) ein.

1. Starte mit *k* beliebigen (zufälligen) Zentren.
2. Weise jeden Punkt dem nächstgelegenen Zentrum zu und bilde so *k* Cluster.
3. Berechne für jeden Cluster seinen Schwerpunkt; das sind die neuen Zentren.
4. Gehe nach 2.