Kapitel ML:IX

IX. Ensemble and Meta

- Ensemble Methods
- Evolutionary Strategies
- Swarm Strategies

Ensemble Methods
Motivation (1): Generalisierungsfähigkeit

- *Err*$^*(y)$ (wahre Missklassifikationsrate)
- *Err*$(y, D_{tr}) = \frac{|\{(\mathbf{x},c(\mathbf{x}))\in D_{tr}:c(\mathbf{x})\neq y(\mathbf{x})\}|}{|D_{tr}|}$ (Missklassifikationsrate auf Trainingsmenge)
- *Err*$(y, D_{ts}) = \frac{|\{(\mathbf{x},c(\mathbf{x}))\in D_{ts}:c(\mathbf{x})\neq y(\mathbf{x})\}|}{|D_{ts}|}$ (Missklassifikationsrate auf Testmenge)

Err(y, D_{ts}) ermöglicht auch den Vergleich von Lernverfahren.

→ Gibt es ein bestes Lernverfahren?

Ensemble Methods

Motivation (2): No Free Lunch Theorem

"There Ain't No Such Thing As A Free Lunch."

[Heinlein, 1966]

In der Optimierung:

"[...] all algorithms that search for an extremum of a cost function perform exactly the same, when averaged over all possible cost functions."

[Wolpert/Macready, 1995]

Im Maschinellen Lernen:

"Some of those theorems show, loosely speaking, that for any two algorithms A and B, there are "as many" targets for which algorithm A has lower expected OTS [off-training set sampling] error than algorithm B as vice-versa (whether one averages over training sets or not)."

[Wolpert 1996]

Ensemble Methods

Motivation (3): Instabilität von Lernverfahren

Heuristische Formulierung:

Ein Lernverfahren heißt instabil, wenn eine kleine Veränderung in den Trainingsdaten eine große Veränderung im gelernten Klassifikator bewirkt.

Instabile Lernverfahren:

- Neuronale Netze (mit fixer Anzahl von hidden Units)
- Entscheidungsbäume (mit fixer Anzahl von Blattknoten)

Stabile Lernverfahren:

- Bayes
- k Nearest Neighbor

Ensemble Methods

Motivation (4): Consequences

- Statistisches Problem.

  Verfahren betrachten eine – gemessen an der Menge von Trainingsdaten – zu große Menge von Hypothesen.

  ➜ Auf Basis der Trainingsdaten eignen sich mehrere Hypothesen gleichermaßen gut als Klassifizierer.

- Rechentechnisches Problem.

  Aufgrund der Komplexität des Problems kann das Lernverfahren nicht das Finden einer besten Lösung innerhalb der Hypothesenmenge garantieren.

  ➜ Bei Verwendung von Heuristiken of nur suboptimale Lösung.

- Repräsentationsproblem:

  Die Kandidatenmenge der Hypothesen enthält keine ausreichend guten Approximationen des Zielkonzeptes.

  ➜ Lernverfahren kann den gewünschten Approximationsgrad nicht liefern.

Ensemble Methods

Bootstrap Aggregating (Bagging)

Idee:

Eine Gruppe von Klassifikatoren, die gemeinsam klassifizieren, kann die Nachteile einzelner Klassifikatoren aufwiegen.

Problem:

Das Lernverfahren braucht verschiedene Trainingsmengen, um verschiedene Klassifikatoren zu bestimmen.

Lösung:

Generierung von ähnlichen Trainingsmengen durch Bootstrapping
(vgl. auch Kreuzvalidierung).

Remarks:

- Aufgrund der untergeordneten Rolle der einzelnen Klassifizierer beim Bagging, besonders aber aufgrund ihrer recht schwachen Klassifikationsvermögens im Kontext des Boosting oder des Cascading, nennen wir diese auch *schwache* Klassifizierer.

Ensemble Methods

Bootstrap Aggregating [Breiman 1994]

- Ausgangspunkt Lernmenge D mit n Beispielen

- Für $t = 1, \ldots, T$:
  Ziehe aus D insgesamt n Beispiele mit Zurücklegen und bilde daraus die Lernmenge D_t

- Mit der Lernmenge $D_t, \ldots, D_T$ werden mit Hilfe eines Lernverfahrens die einzelne Klassifikatoren y_t bestimmt.

- Die Klassifikatoren $y_1, \ldots, y_T$ werden zu einem Ensemble zusammengefasst und legen durch Mehrheitsentscheid die Klasse eines Beispiels fest:

$$y(x) := \underset{j \in \{1,\ldots,J\}}{\text{argmax}} \left|\{t \in \{1, \ldots, T\} : y_t(x) = j\}\right|$$

Ensemble Methods

Bootstrap Aggregating (Fortsetzung)

Algorithm: Bagging

Input: Lernbeispiele $D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))\}$, $n \in \mathbf{N}$
mit $\mathbf{x}_i \in X$ und $c(\mathbf{x}_i) \in \{1, \ldots, J\}$ für $1 \leq i \leq n$;
Anzahl T mit $T \in \mathbf{N}$ für die Anzahl der Klassifizierer.

Output: Klassifizierer y für X.

1. Für $t = 1, \ldots, T$:
   (a) Bestimme Lernmenge D_t durch Bootstrapping aus D
   (Ziehen von n Beispielen aus D mit Zurücklegen)
   (b) Trainiere einen schwachen Klassifikator y_t, d.h. $y_t : X \to \mathbf{R}$ mit der Beispielmenge D_t.
2. Ergebnis ist der Klassifikator

$$y(x) := \underset{j \in \{1,\ldots,J\}}{\text{argmax}} \left| \{t \in \{1, \ldots, T\} : y_t(x) = j\} \right|$$

Ensemble Methods

Bootstrap Aggregating (Fortsetzung)

- Die Wahrscheinlichkeit, dass ein Beispiel mindestens einmal gezogen wird, ist $1 - (1 - 1/n)^n$.
- Für n groß, gilt $1 - (1 - 1/n)^n \approx 1 - 1/e \approx 0.632$.
- In jeder Lernmenge sind etwa 63.2% der Beispiele in D.
- Verbesserungen der Fehlerrate von 20% bis 47% bei Anwendung mit Entscheidungsbäumen wurden beobachtet.

Ensemble Methods

Boosting Weak Classifiers [Freund/Schapire 1995]

Idee:

Eine Gruppe von Klassifikatoren, die gemeinsam klassifizieren, kann die Nachteile einzelner Klassifikatoren aufwiegen.

Problem:

Das Lernverfahren braucht verschiedene Trainingsmengen, um verschiedene Klassifikatoren zu bestimmen.

Lösung:

Verschränkung von Lernalgorithmus und Generierung von Lernmengen:
Gewichtung der Lernbeispiele (Änderung der relativen Häufigkeiten) aufgrund der Auswertung des vorherigen Klassifikators.

Ensemble Methods

Adaptive Boosting (AdaBoost) [Freund/Schapire 1995]

- Ausgangspunkt Lernmenge D mit n Beispielen
- Gewichtung der Lernbeispiele entsprechend dem Klassifikationsergebnis des zuletzt generierten “schwachen” Klassifikators
  - Verringerung des Gewichts von korrekt klassifizierten Beispielen
  - Erhöhung des Gewichts von falsch klassifizierten Beispielen
- Mit der neuen Lernmenge wird mit Hilfe eines Lernverfahrens der nächste Klassifikator bestimmt.
- Die Klassifikatoren $y_1, \ldots, y_T$ werden zu einem Ensemble zusammengefasst und legen durch gewichteten Mehrheitsentscheid die Klasse eines Beispiels fest.

➜ Anwendung z.B. mit teilweise gelernten Entscheidungsbäumen

Ensemble Methods

Adaptive Boosting (Fortsetzung)

Algorithm: AdaBoost.M1

Input: Lernbeispiele $(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))$, $n \in \mathbf{N}$
mit $\mathbf{x}_i \in X$ und $c(\mathbf{x}_i) \in \{1, \ldots, J\}$ für $1 \leq i \leq n$;
Anzahl T mit $T \in \mathbf{N}$ für die Anzahl der Runden.

Output: Klassifizierer y für X.

1. Initialisiere Gewichte für alle Beispiele durch $w_1(i) = 1/n$ für $1 \leq i \leq n$.
2. Für $t = 1, \ldots, T$ führe folgende Schritte aus:
   (a) Trainiere einen schwachen Klassifikator y_t, d.h. $y_t : X \to \mathbf{R}$, mit den durch w_t gewichteten Beispielen.
   (b) Sei $\varepsilon_t = \sum_{i=1}^{n} w_t(i) \cdot (1 - \delta(y_t(\mathbf{x}_i), c(\mathbf{x}_i)) = \sum_{\{i \mid y_t(\mathbf{x}_i) \neq c(\mathbf{x}_i)\}} w_t(i)$.
   (δ Kronecker-Funktion, d.h. $\delta(u, v) = 1$ für $u = v$ und $\delta(u, v) = 0$ sonst.)
   (c) Setze $\beta_t = \frac{\varepsilon_t}{(1-\varepsilon_t)}$.
   (d) Setze $w_{t+1}(i) = \begin{cases} w_t(i) \cdot \beta_t \cdot 1/z_t & \text{falls } y_t(\mathbf{x}_i) = c(\mathbf{x}_i) \\ w_t(i) \cdot 1/z_t & \text{sonst} \end{cases}$ für $1 \leq i \leq n$.
   z_t ist Normalisierungsfaktor, durch den das Gesamtgewicht aller Beispiele den Wert 1 erhält, also eine Verteilung widerspiegelt.
3. Ergebnis ist der Klassifikator $y(\mathbf{x}) = \underset{j \in \{1,\ldots,J\}}{\text{argmax}} \sum_{\{t \mid y_t(\mathbf{x}) = j\}} \log \dfrac{1}{\beta_t}$

Ensemble Methods

Adaptive Boosting (Fortsetzung)

Der Klassifikator y gewichtet die Entscheidungen der einzelnen Klassifikatoren stärker, wenn ihr Fehler klein ist.

Wenn die einzelnen Klassifikatoren eine bessere Fehlerrate als $1/2$ haben, dann fällt der Fehler von y exponentiell in T gegen 0.

Satz 1

Falls für die Fehlerraten ε_t während des Ablaufs von Algorithmus AdaBoost.M1 gilt $\varepsilon_t \leq 1/2$, so folgt für den trainierten Klassifizierer y

$$\frac{1}{n} \cdot |\{i : y(\mathbf{x}_i) \neq c(\mathbf{x}_i)\}| \leq \text{exp}\left(-2 \sum_{t=1}^{T} \left(\frac{1}{2} - \varepsilon_t\right)^2\right)$$

Remarks:

- Problem: Fehlerrate der im Fall von J Klassen mit $J > 2$ nicht so einfach erreichbar.
- Spezialfall $j = 2$: Klassifikationsproblem mit genau 2 Klassen, ein schwacher Klassifizierer muss nur geringfügig besser sein als Raten.
- Betrachtung der Klassen $\{-1, +1\}$ erlaubt einfachere Schreibweisen.

Ensemble Methods

Discrete AdaBoost

Algorithm: Discrete AdaBoost

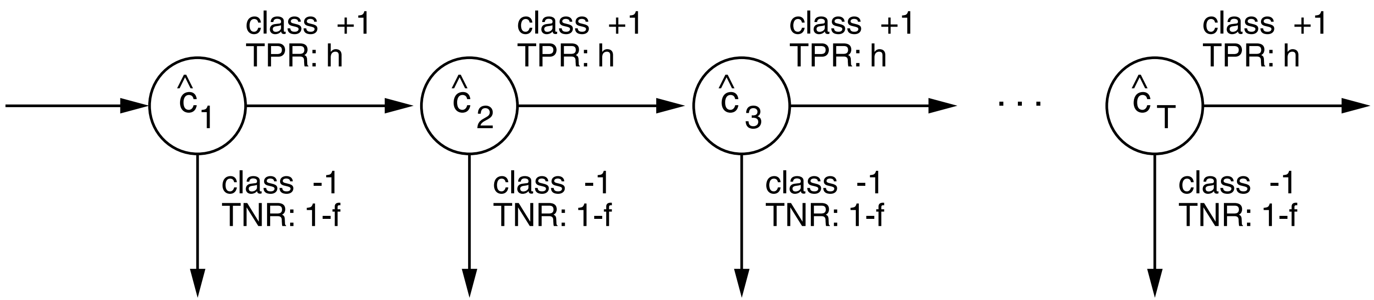
Input: Lernbeispiele $(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))$, $n \in \mathbf{N}$
mit $\mathbf{x}_i \in X$ und $c(\mathbf{x}_i) \in \{-1, +1\}$ für $1 \leq i \leq n$;
Anzahl T mit $T \in \mathbf{N}$ für die Anzahl der Runden.

Output: Klassifizierer y für X.

1. Initialisiere Gewichte für alle Beispiele durch $w_1(i) = 1/n$ für $1 \leq i \leq n$.
2. Für $t = 1, \ldots, T$ führe folgende Schritte aus:
   (a) Trainiere einen schwachen Klassifikator c_t, d.h. $y_t : X \to \{-1, +1\}$, mit den durch w_t gewichteten Beispielen.
   (b) Sei $\varepsilon_t = \sum_{i=1}^{n} w_t(i) \cdot 1/2 \cdot |y_t(\mathbf{x}_i) - c(\mathbf{x}_i)| = \sum_{\{i \mid y_t(\mathbf{x}_i) \neq c(\mathbf{x}_i)\}} w_t(i)$.
   (c) Setze $\beta_t = \frac{\varepsilon_t}{(1-\varepsilon_t)}$.
   (d) Setze $w_{t+1}(i) = \begin{cases} w_t(i) \cdot \beta_t \cdot 1/z_t & \text{falls } y_t(\mathbf{x}_i) = c(\mathbf{x}_i) \\ w_t(i) \cdot 1/z_t & \text{sonst} \end{cases}$ für $1 \leq i \leq n$.
   z_t ist Normalisierungsfaktor, durch den das Gesamtgewicht aller Beispiele den Wert 1 erhält, also eine Verteilung widerspiegelt.
3. Ergebnis ist der Klassifikator $y(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} \log \frac{1}{\beta_t} \cdot y_t(\mathbf{x})\right)$

Ensemble Methods

Cascades of Classifiers (Cascading) [Viola/Jones 2001]



- Gesucht wird eine Folge $y_1, \ldots, y_T$ von Klassifikatoren mit steigender Komplexität für ein 2-Klassen-Problem.
- In der Lernmenge D überwiegen die Negativbeispiele (Klasse -1).
- Für jeden Klassifikator werden Mindestbedingungen gestellt an

  die *Hitrate* $\frac{|\{\mathbf{x} \in D : y_t(x) = c(\mathbf{x}) = +1\}|}{|\{\mathbf{x} \in D : c(\mathbf{x}) = +1\}|} \geq h$ und die *False Alarm Rate* $\frac{|\{\mathbf{x} \in D : y_t(\mathbf{x}) = +1,\ c(\mathbf{x}) = -1\}|}{|\{x \in D : c(\mathbf{x}) = -1\}|} < f$.
- Aus der Lernmenge D wird eine Teilmenge D_t von Beispielen gezogen, die von der bisherigen Kaskade mit $+1$ klassifiziert werden. Das Verhältnis von Positivbeispielen und Negativbeispielen in D_t ist fest.
- Mit D_t wird durch Boosting ein Ensemble y_t von schwachen Klassifizierern gelernt.
- Für 10 Stufen in der Kaskade, einer Hitrate von mindestens 0.99 und einer False Alarm Rate von höchstens 0.3 erhält man für die Kaskade eine Hitrate von $0.99^{10} \approx 0.9$ und eine False Alarm Rate von höchstens $0.3^{10} \approx 0.000006$.

Ensemble Methods

Cascading (Fortsetzung)

In einem 2-Klassen-Problem wird eine Lern- oder Testmenge D durch einen Klassifikator c in vier disjunkte Teilmengen aufgeteilt:

1. D_h Menge der Beispiele $\mathbf{x}$ mit Klasse $+1$ und $y(\mathbf{x}) = +1$ (Index h steht für *hits*, Menge der richtig positiven Beispiele, true positive),
2. D_f Menge der Beispiele $\mathbf{x}$ mit Klasse -1 und $y(\mathbf{x}) = +1$ (Index f steht für *false alarms*, Menge der falsch positiven Beispiele, false positive, Fehler 1. Art),
3. D_m Menge der Beispiele $\mathbf{x}$ mit Klasse $+1$ und $y(\mathbf{x}) = -1$ (Index m steht für *misses*, Menge der falsch negativen Beispiele, false negative, Fehler 2. Art),
4. D_z Menge der Beispiele $\mathbf{x}$ mit Klasse -1 und $y(\mathbf{x}) = -1$ (Index z steht für *zero*, Menge der richtig negativen Beispiele, true negative).

(*richtig/falsch* bezieht sich auf Übereinstimmung von $c(\mathbf{x})$ und $y(\mathbf{x})$, *positiv/negativ* auf die Klassifikation durch y, also $y(\mathbf{x})$.) Diese Mengen werden häufig in einer Tabelle angeordnet:

	$c = +1$	$c = -1$	
$y = +1$	D_h	D_f	D_{pPos}
$y = -1$	D_m	D_z	D_{pNeg}
	D_{Pos}	D_{Neg}	D

D_{pNeg} und D_{Neg} bezeichnen die Mengen der (predicted) Negatives, D_{pPos} und D_{Pos} bezeichnen die Mengen der (predicted) Positives.

Ensemble Methods

Maßzahlen für Cascading

Mit den Kardinalitäten dieser Mengen lassen sich verschiedene Maßzahlen definieren. Maßzahlen zur Beurteilung des Tests:

- *TPR* $= P(y(\mathbf{x}) = +1 | c(\mathbf{x}) = +1) \approx \frac{|D_h|}{|D_h|+|D_m|} = \frac{|D_h|}{|D_{Pos}|}$
  Wahrscheinlichkeit einer richtig positiven Vorhersage
  (Richtigpositiv-Rate, True Positive Rate, Trefferquote, Hit Rate, Probability of Detection, Sensitivity, Recall, Sensitivität, Empfindlichkeit eines Tests)

- *FNR* $= P(y(\mathbf{x}) = -1 | c(\mathbf{x}) = +1) \approx \frac{|D_m|}{|D_h|+|D_m|} = \frac{|D_m|}{|D_{Pos}|}$
  Wahrscheinlichkeit einer falsch negativen Vorhersage
  (Falschnegativ-Rate, False Negative Rate, Pendant zur Sensitivität, $1-$ *TPR*)

- *TNR* $= P(y(\mathbf{x}) = -1 | c(\mathbf{x}) = -1) \approx \frac{|D_f|}{|D_f|+|D_z|} = \frac{|D_f|}{|D_{Neg}|}$
  Wahrscheinlichkeit einer richtig negativen Vorhersage
  (Richtignegativ-Rate, Correct Rejection Rate, True Negative Rate, Spezifität)

- *FPR* $= P(y(\mathbf{x}) = +1 | c(\mathbf{x}) = -1) \approx \frac{|D_f|}{|D_f|+|D_z|} = \frac{|D_f|}{|D_{Neg}|}$
  Wahrscheinlichkeit eines falschen Alarms
  (Falschpositiv-Rate, False Alarm Rate, False Positive Rate, Probability of False Detection, Pendant zur Spezifität, $1-$ *TNR*)

Ensemble Methods

Maßzahlen für Cascading (Fortsetzung)

Maßzahlen zur Beurteilung der Population:

- *PPV* $= P(c(\mathbf{x}) = +1 | y(\mathbf{x}) = +1) \approx \frac{|D_h|}{|D_h|+|D_f|} = \frac{|D_h|}{|D_{pPos}|}$
  Wahrscheinlichkeit der Korrektheit bei positiver Vorhersage
  (positiv prädiktiver Wert, Positive Predictive Value, Frequency of Hits, Precision, Relevanz)

- *NPV* $= P(c(\mathbf{x}) = -1 | y(\mathbf{x}) = -1) \approx \frac{|D_z|}{|D_m|+|D_z|} = \frac{|D_z|}{|D_{pNeg}|}$
  Wahrscheinlichkeit der Korrektheit bei negativer Vorhersage
  (negativ prädiktiver Wert, Negative Predictive Value, Segreganz, Trennfähigkeit)

- *FAR* $= P(c(\mathbf{x}) = -1 | y(\mathbf{x}) = +1) \approx \frac{|D_f|}{|D_h|+|D_f|} = \frac{|D_f|}{|D_{pPos}|}$
  Wahrscheinlichkeit der Nicht-Korrektheit bei positiver Vorhersage
  (False Alarm Ratio)

- *CCR* $\approx \frac{|D_h|+|D_z|}{|D_h|+|D_f|+|D_m|+|D_z|} = \frac{|D_h|+|D_z|}{|D|}$
  Wahrscheinlichkeit einer korrekten Klassifikation
  (Korrektklassifikationsrate, Correct Classification Rate)

- *FCR* $\approx \frac{|D_f|+|D_m|}{|D_h|+|D_f|+|D_m|+|D_z|} = \frac{|D_f|+|D_m|}{|D|}$
  Wahrscheinlichkeit einer falschen Klassifikation
  (Falschklassifikationsrate, False Classification Rate)

Ensemble Methods

Cascading (Fortsetzung)

Algorithm: Cascading

Input: Lernbeispiele $D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_N, c(\mathbf{x}_n))\}$, $n \in \mathbf{N}$
mit $\mathbf{x}_i \in X$ und $c(\mathbf{x}_i) \in \{+1, -1\}$ für $1 \leq i \leq n$;
Anzahl T mit $T \in \mathbf{N}$ für die Anzahl der Stufen.

Output: Klassifizierer y für X.

1. Für $t = 1, \ldots, T$ führe folgende Schritte aus:
   (a) Bestimme Lernmenge D_t durch Ziehen von n' Beispielen aus D, die unter der bisherigen Kaskade als $+1$ klassifiziert wurden.
   (b) Trainiere einen Klassifikator y_t, d.h. $y_t : X \to \mathbf{R}$ mit der Beispielmenge D_t.
2. Ergebnis ist der Klassifikator

$$y(x) := \begin{cases} +1 & \text{falls } y_t(x) = +1 \text{ für alle } t \in \{1, \ldots, T\} \\ -1 & \text{sonst} \end{cases}$$

Ensemble Methods

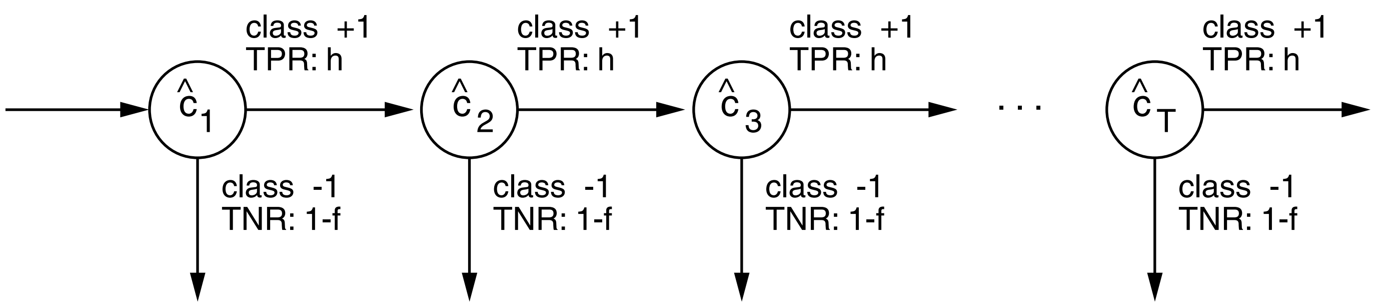
Cascading for Face Detection (Main Loop)

- Graustufenbilder
- Suchbereich mit fester Größe 20×20 Pixel
- verschiedene Skalierungsstufen des Ausgangsbildes: Faktor $1/1.1 \approx 0.91$
- zeilenweises Überstreichen mit Suchbereich
- Überprüfen des Ausschnitts auf Gesicht

Ensemble Methods

Cascading for Face Detection (Gesamtklassifikator)

- Kaskade mit $T = 22$
- False Alarm Rate pro Stufe: *FPR* $= 0.5$
- True Positive Rate pro Stufe: *TPR* $= 0.999$
- False Alarm Rate gesamt: $\approx 0.5^{22} \approx 0.00000024$
- True Positive Rate Rate gesamt: $\approx 0.999^{22} \approx 0.978$
- Trainingsbeispiele für Stage $i+1$ sind positiv klassifizierte Beispiele der Stage i



Ensemble Methods

Cascading for Face Detection(Einzelklassifikator in Kaskade)

- Ensemble-Klassifizierer mit Discrete AdaBoost bestimmen
- Stage 1: drei schwache Klassifikatoren ... Stage 22: 213 schwache Klassifikatoren

Algorithm: Discrete AdaBoost

Input: Lernbeispiele $(\mathbf{x}_1, c(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, c(\mathbf{x}_n))$, $n \in \mathbf{N}$
mit $\mathbf{x}_i \in X$ und $c(\mathbf{x}_i) \in \{-1, +1\}$ für $1 \leq i \leq n$;
Anzahl T mit $T \in \mathbf{N}$ für die Anzahl der Runden.

Output: Klassifizierer y für X.

1. Initialisiere Gewichte für alle Beispiele durch $w_1(i) = 1/n$ für $1 \leq i \leq n$.
2. Für $t = 1, \ldots, T$ führe folgende Schritte aus:
   (a) Trainiere einen schwachen Klassifikator c_t, d.h. $y_t : X \to \{-1, +1\}$, mit den durch w_t gewichteten Beispielen.
   (b) Sei $\varepsilon_t = \sum_{i=1}^{n} w_t(i) \cdot 1/2 \cdot |y_t(\mathbf{x}_i) - c(\mathbf{x}_i)| = \sum_{\{i | y_t(\mathbf{x}_i) \neq c(\mathbf{x}_i)\}} w_t(i)$.
   (c) Setze $\beta_t = \frac{\varepsilon_t}{(1-\varepsilon_t)}$.
   (d) Setze $w_{t+1}(i) = \begin{cases} w_t(i) \cdot \beta_t \cdot 1/z_t & \text{falls } y_t(\mathbf{x}_i) = y_i \\ w_t(i) \cdot 1/z_t & \text{sonst} \end{cases}$ für $1 \leq i \leq n$.
   z_t ist Normalisierungsfaktor, durch den das Gesamtgewicht aller Beispiele den Wert 1 erhält, also eine Verteilung widerspiegelt.
3. Ergebnis ist der Klassifikator

$$y(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} \log \frac{1}{\beta_t} \cdot y_t(\mathbf{x})\right)$$

Ensemble Methods

Cascading for Face Detection (Einzelklassifikator in Ensemble)

- Schwacher Klassifikator: CART Klassifikator mit genau einem Knoten
- Gesucht: Threshold mit bestem Missklassifikationsgewicht für Feature
- Auswahl: Merkmal mit bestem Missklassifikationsgewicht
- Größe der Trainingsmenge: ca. 5000 Bespiele für Gesichter, ca. 3000 Beispiele ohne Gesichter pro Stufe

→ Welche Merkmale verwendet man für die Gesichtserkennung?

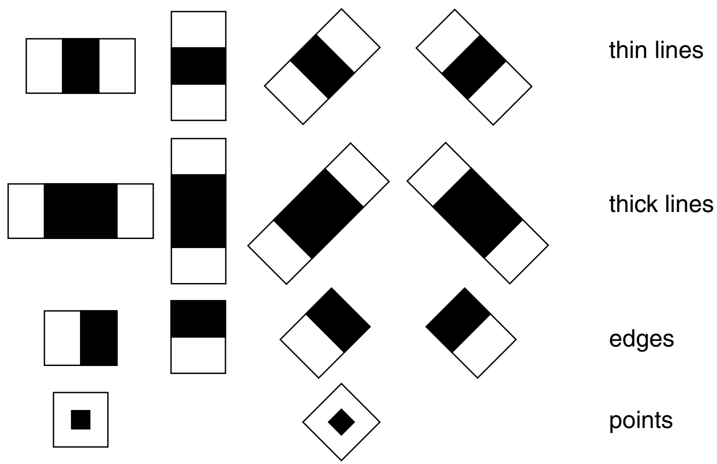
Ensemble Methods

Cascading for Face Detection (alle Features)

- Verwendung der 20×20 Bildpunkte, also 400 Features mit je 256 Werten

➜ Schwache Klassifikatoren verwenden einzelne Bildpunkte.

Aussagefähigere Merkmale:



- Merkmale sind Helligkeitsunterschiede zwischen Flächen.
- Merkmale: Typ $\times$ Position $\times$ Breite $\times$ Höhe
- Merkmale: Anzahl verschiedener Merkmale > 100.000

Ensemble Methods

Cascading for Face Detection (ausgewählte Features)

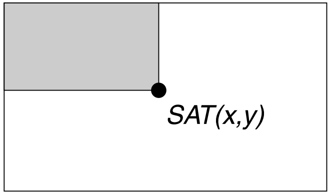
Haben die ausgewählten Merkmale eine Bedeutung für uns?

- Zusammenfassung der Merkmale der Stufen 1 bis 11
- Auswertung der Merkmale:
  - Ausgangspunkt graue Fläche (Wert 127)
  - dunkler Bereich des Merkmals verringert Farbwerte
  - heller Bereich des Merkmals erhöht Farbwerte
  - Betrag abhängig vom Threshold
- Nicht alle Merkmale sind anzeigbar.

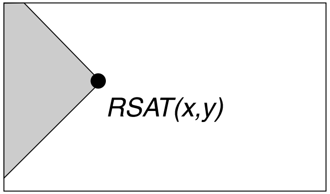
Ensemble Methods

Cascading for Face Detection (Effizienz)

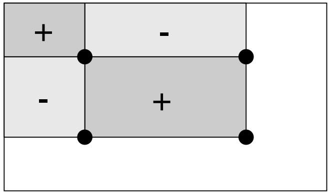
Wie lassen sich die Merkmale effizient berechnen?



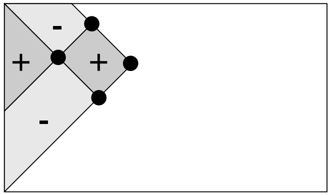
SummedAreaTable(x,y)



RotatedSummedAreaTable(x,y)



Area pixel sum



Rotated area pixel sum

- $\text{SAT}(x,y) = \text{SAT}(x,y-1) + \text{SAT}(x-1,y) + I(x,y) - \text{SAT}(x-1,y-1)$
  mit $\text{SAT}(-1,y) = \text{SAT}(x,-1) = 0$
- Pass 1:
  $\text{RSAT}(x,y) = \text{RSAT}(x-1,y-1) + \text{RSAT}(x-1,y) + I(x,y) - \text{RSAT}(x-2,y-1)$
  mit $\text{RSAT}(-1,y) = \text{RSAT}(-2,y) = \text{RSAT}(x,-1) = 0$
- Pass 2:
  $\text{RSAT}(x,y) = \text{RSAT}(x,y) + \text{RSAT}(x-1,y+1) - \text{RSAT}(x-2,y)$

Ensemble Methods

Cascading for Face Detection (Effizienz beim Training)

- Anzahl Trainingsbeispiele: > 10.000
- Anzahl Merkmale: > 100.000
- Benötigter Hauptspeicher: 4 Byte $\times$ 10.000 $\times$ 100.000 > 3,7GByte

➜ Merkmalwerte werden (zeilenweise) immer wieder neu berechnet. Mehrere Tage Trainingslaufzeit.

Ensemble Methods

Trainingsmengen

- Bagging
  D_t wird aus D_t gebildet durch Ziehen mit Zurücklegen.
  ➜ Trainingsmengen voneinander unabhängig.

- Boosting
  D_t wird aus D_{t-1} gebildet durch stärkere Gewichtung falsch klassifizierter Beispiele.
  ➜ Trainingsmengen bauen aufeinander auf, sind aber bis auf Gewichtung gleich.

- Cascading
  D_t wird aus D_{t-1} gebildet durch Beschränkung auf korrekt klassifizierte Beispiele.
  ➜ Trainingsmengen bauen aufeinander auf, sind jeweils Teilmengen der vorherigen.

Ensemble Methods

Grundlage der Entscheidung

- Bagging
  Einfache Mehrheitsentscheidung durch schwache Klassifizierer.

- Boosting
  Gewichtete Mehrheitsentscheidung durch schwache Klassifizierer.

- Cascading
  Sequentielle Entscheidung durch schwache Klassifizierer: $+1$ nur bei Einstimmigkeit.

Ensemble Methods

Literature

- P. Viola, M. Jones.
  Rapid Object Detection using a Boosted Cascade of Simple Features
- R. Lienhart, J. Maydt.
  An Extended Set of Haar-like Features for Rapid Object Detection
- R. Lienhart, A. Kuranov, V. Pisarevsky.
  Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection
- R. Lienhart, L. Liang, A. Kuranov.
  A Detector Tree of Boosted Classifiers for Real-time Object Detection and Tracking

