Von schwachen zu starken Lernern

Wir nehmen an, dass ein schwacher Lernalgorithmus L mit vielen Beispielen, aber großem Fehler $\varepsilon = \frac{1}{2} - \theta$ gegeben ist.

- Wie lässt sich der Verallgemeinerungsfehler ε von L signifikant senken, ohne dass neue Beispiele angefordert werden?

  Allgemeine Lernverfahren lassen sich nicht mit Garantie verbessern, PAC-Algorithmen aber lassen sich stets verbessern.
- Wir besprechen zwei Verfahren Bagging and Boosting, die durch Neugewichtung der Beispiele neue Hypothesen $h_1, \ldots, h_k$ erzeugen.
- Eine endgültige Hypothese wird dann nach Mehrheitsbildung ausgegeben.

Bagging

Eingabe: Ein Lernalgorithmus L und eine Menge B von s klassifizierten Beispielen. k sei eine ungerade Zahl.

(1) For $t = 1$ to k do

(1a) Wähle s klassifizierte Beispiele, mit Ersetzung, aus der Menge B. /* Ein gewähltes Beispiel ist zurückzulegen. Es können also Beispiele mehrfach gezogen werden. Im Durchschnitt werden 63.2% der Beispiele aus B gewählt. */

(1b) Wende den Algorithmus L auf die gewählten Beispiele an, um die Hypothese h_t zu erhalten.

(2) Gib die Mehrheitshypothese h aus: Es ist $h(x) = 1$ (bzw $h(x) = 0$) genau dann, wenn eine Mehrheit der Hypothesen $h_1, \ldots h_k$ auf x die Klassifizierung 1 (bzw. 0) liefert.

Verbesserungsleistung von Bagging

(1) Bagging hilft, wenn der Lernalgorithmus instabil ist, also auf kleine Änderungen in der Beispielsmenge mit stark modifizierten Hypothesen reagiert:

    Entscheidungsbaum-Algorithmen und neuronale Netzwerke sind Beispiele solch instabiler Lernverfahren.

(2) Überraschende Erkenntnis: Oft droht sogar für grosse Werte von k kein Overfitting.

    Wir werden dasselbe Phänomen auch für Boosting antreffen und dort erklären.

Das Setup

- Ein PAC-Algorithmus L löst ein binäres Klassifikationsproblem mit der nur sehr schwachen Fehlerschranke $\varepsilon = \frac{1}{2} - \theta$.
- Die Hypothesen von L klassifizieren die Beispiele mit 1 und -1.

(1) Zuerst wird eine entsprechend große Menge $S = \{(x_1, b_1), \ldots, (x_s, b_s)\}$ klassifizierter Beispiele angefordert.

(2) Wenn L eine konsistente Hypothese h_1 bestimmt, sind wir fertig. Sonst:

- Wir erhöhen das Gewicht falsch klassifizierter Beispiele auf 50% und fordern L auf, eine neue Hypothese h_2 zu liefern.
- Ist h_2 inkonsistent, setze Beispielgewichte neu: Falsch klassifizierter Beispiele erhalten wiederum das Gewicht 50%.
- Wir wiederholen dieses Vorgehen genügend oft.

(3) Wir geben „so etwas“ wie eine Mehrheitshypothese aus.

AdaBoost: Adaptive Boosting

Eingabe: Ein Lernalgorithmus L und s klassifizierte Beispiele $(x_1, b_1), \ldots, (x_s, b_s)$.

(1) Sei D_1 die Gleichverteilung: Setze $w_i = 1$ für $i = 1, \ldots, s$.

(2) For $t = 1$ to T do

(2a) L bestimmt eine Hypothese h_t mit „Trainingsfehler“

$$\varepsilon_t = \text{prob}_{D_t}[\, h_t(x_i) \neq b_i \,].$$

(2b) Setze $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$ und aktualisiere $w_i = w_i \cdot \beta_t$ für alle durch h_t richtig klassifizierten Beispiele (x_i, b_i).
/* Das Gewicht richtig klassifizierten Beispiele ist $(1 - \varepsilon_t) \cdot \beta_t = \varepsilon_t$ und gleich dem Gewicht falsch klassifizierten Beispiele. */

(2c) Die neue Verteilung D_{t+1} wird durch $\text{prob}[x_i] = \frac{w_i}{\sum_{i=1}^{s} w_i}$ definiert.

(3) Setze $\alpha_t = \ln(\frac{1}{\beta_t})$ und gib die Hypothese $\text{sign}(\sum_{t=1}^{T} \alpha_t h_t)$ aus.
/* Je größer α_t, je kleiner ist der Trainingsfehler ε_t von h_t. */

Experimentelle Ergebnisse

In typischen Anwendungen liegen aber keine PAC-Algorithmen vor!

(1) Wie Bagging erreicht AdaBoost häufig eine Verbesserung, wenn das Lernverfahren auf relativ kleine Veränderungen der Beispielmenge mit einer stark veränderten Hypothese reagiert.

(2) Overfitting tritt nicht auf:

    Es gibt sogar Fälle, in denen der Verallgemeinerungsfehler weiter fällt, obwohl der Trainingsfehler bereits auf Null gesenkt wurde.

(3) AdaBoost scheint durchweg bessere Ergebnisse als Bagging zu liefern.

Boosting für neuronale Netzwerke: LeNet4

LeNet3 ist ein neuronales Netz zur Ziffernerkennung mit einer Fehlerrate von 1%.
LeNet4 führt eine Boosting Variante mit $T = 3$ durch.

(1) Ein erstes neuronales Netz wird konventionell trainiert.

(2) Ein zweites Netz erhält zur Hälfte vom ersten Netz falsch klassifizierte und zur anderen Hälfte richtig klassifizierte Ziffern.

(3) Das dritte Netz erhält als Trainingsmenge alle von den ersten beiden Netzen falsch klassifizierten Ziffern.

(4) Das erste Netz ist sehr gut: Um genügend viele falsch klassifizierte Beispiele zu erhalten, vergrößere die Trainingsmenge durch affine Transformationen und Änderungen der Schriftbreite.

Das „3-er Ensemble“ erreichte eine Fehlerrate von ungefähr 0.7%.

Analyse von AdaBoost: Ein Überblick

Die Analyse wird in zwei Schritten durchgeführt:

(1) Zuerst zeigen wir, dass die Mehrheits-Hypothese sign(h) einen sehr kleinen Trainingsfehler besitzt,

    wenn der von AdaBoost benutzte Lernalgorithmus L auf jeder Verteilung D_t einen nicht zu grossen Fehler ε_t besitzt.

(2) Dann weisen wir nach, dass ein kleiner Trainingsfehler zu einem kleinen Verallgemeinerungsfehler führt.

    Insbesondere erhalten wir, dass Overfitting selbst für eine relativ große Zahl T von Hypothesen nicht eintritt.

AdaBoost und Weighted Majority

AdaBoost kann als eine randomisierte Version von Weighted Majority angesehen werden:

- Die Beispiele x_i übernehmen die Rolle der Experten.
- Die Gewichtssetzung favorisiert informative Beispiele, also Beispiele, die häufig falsch klassifiziert werden.

Wir übernehmen deshalb auch Ideen aus der Analyse von Weighted Majority:

- Wir bestimmen untere und obere Schranken für das Gesamtgewicht W_t zum Zeitpunkt t.
- $W_1 = s$.
- Nach jedem Zeitpunkt t: Das Gesamtgewicht richtig klassifizierten Beispiele ist gleich $\varepsilon_t \cdot W_t$ und stimmt mit dem Gewicht falsch klassifizierten Beispiele überein.

Als Konsequenz: $W_{t+1} = 2\varepsilon_t \cdot W_t$.

AdaBoost: Der Trainingserfolg I

Wann klassifiziert AdaBoost das Beispiel x_i falsch?

- x_i werde zu Zeitpunkten in $I_r \subseteq \{1, \ldots, T\}$ richtig und zu Zeitpunkten in $I_f \subseteq \{1, \ldots, T\}$ falsch klassifiziert.
- Es gilt für $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$:

$$\sum_{t \in I_r} \ln(\frac{1}{\beta_t}) \leq \sum_{t \in I_f} \ln(\frac{1}{\beta_t})$$

- Wir exponentieren beide Seiten

$$\Pi_{t \in I_r} \frac{1}{\beta_t} = e^{\sum_{t \in I_r} \ln(\frac{1}{\beta_t})} \leq e^{\sum_{t \in I_f} \ln(\frac{1}{\beta_t})} = \Pi_{t \in I_f} \frac{1}{\beta_t}.$$

Wenn AdaBoost x_i falsch klassifiziert, dann ist $\Pi_{t \in I_f} \beta_t \leq \Pi_{t \in I_r} \beta_t$, bzw. $\Pi_{t=1}^{T} \sqrt{\beta_t} \leq \Pi_{t \in I_r} \beta_t$.

AdaBoost: Der Trainingserfolg II

Als Konsequenz einer falschen Klassifizierung folgt $\Pi_{t=1}^{T}\sqrt{\beta_t} \leq \Pi_{t\in I_r}\beta_t$.

(1) Das endgültige Gewicht w_i^{T+1} von x_i stimmt mit $\Pi_{t\in I_r}\beta_t$ überein, da wir mit β_t multiplizieren, wenn Hypothese h_t richtig klassifiziert.

(2) Wenn AdaBoost das Beispiel x_i falsch klassifiziert, dann muss das endgültige Gewicht w_i^{T+1} groß sein, denn

$$w_i^{T+1} = \Pi_{t\in I_r}\beta_t \geq \Pi_{t=1}^{T}\sqrt{\beta_t}.$$

Wenn AdaBoost einen Fehler μ auf der Trainingsmenge S hat, dann folgt

$$\sum_{i=1}^{s} w_i^{T+1} \geq \sum_{i,\text{AdaBoost irrt auf } x_i} w_i^{T+1} \geq \mu s \cdot \Pi_{t=1}^{T}\sqrt{\beta_t}$$

AdaBoost: Der Trainingserfolg III

Je größer der Fehler μ auf der Trainingsmenge, umso größer ist die Gewichtssume W_{T+1} am Ende der Berechnung.

Aber wie groß kann die Gewichtssumme W_{T+1} werden?

(1) Wir haben bereits festgestellt, dass $W_{t+1} = 2\varepsilon_t \cdot W_t$ gilt. Also folgt

$$W_{T+1} = s \cdot \Pi_{t=1}^{T}(2\varepsilon_t),$$

denn $W_1 = s$.

(2) Bei einem großen Fehler μ auf der Trainingsmenge folgt:

$$\mu s \cdot \Pi_{t=1}^{T} \sqrt{\beta_t} \leq s \cdot \Pi_{t=1}^{T}(2\varepsilon_t),$$

$$\text{bzw.} \quad \mu \leq \Pi_{t=1}^{T} \frac{2\varepsilon_t}{\sqrt{\beta_t}} = \Pi_{t=1}^{T} 2\sqrt{\varepsilon_t \cdot (1-\varepsilon_t)},$$

denn $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$.

Der Trainingserfolg: Zusammenfassung

Die Hypothesen $h_1, \ldots, h_T$ mögen die Fehler $\varepsilon_1, \ldots, \varepsilon_T$ haben. Dann hat AdaBoost einen Fehler von höchstens

$$2^T \cdot \Pi_{t=1}^T \sqrt{\varepsilon_t \cdot (1 - \varepsilon_t)}.$$

(1) Falls $\varepsilon_t \leq \varepsilon = 1/2 - \theta$ ist

$$\begin{aligned} & 2^T \cdot \Pi_{t=1}^T \sqrt{\varepsilon_t \cdot (1 - \varepsilon_t)} \leq 2^T \cdot \sqrt{\varepsilon \cdot (1 - \varepsilon)^T} \\ = \; & 2^T \cdot [(1/2 - \theta) \cdot (1/2 + \theta)]^{T/2} = [1 - 4 \cdot \theta^2]^{T/2}. \end{aligned}$$

(2) Der Fehler von AdaBoost konvergiert schnell gegen Null.

(1) Nur richtig klassifizierte Beispiele werden herabgesetzt: W_{T+1} ist groß, wenn der endgültige Trainingsfehler groß ist.

(2) Bei nicht zu großen individuellen Fehlern ε_t ist W_{T+1} aber klein.

AdaBoost: Der Verallgemeinerungsfehler I

Wir können bei genügend großem T sogar Konsistenz garantieren, aber unsere Hypothesenklasse ist komplexer geworden:

Statt mit der ursprünglichen Hypothesenklasse $\mathcal{H}$ arbeiten wir jetzt mit

$$\mathcal{H}_T = \{\text{sign}(\sum_{i=1}^{T} \alpha_i h_i) \mid h_1, \ldots, h_T \in \mathcal{H}, \alpha \geq 0\}.$$

(1) Die VC-Dimension kann signifikant ansteigen! Warum gutartiges Verhalten in der Praxis?

(2) Der Margin steigt mit wachsendem T!

AdaBoost: Der Verallgemeinerungsfehler II

Die Klassifizierung f sei zu lernen. Für Beispiel x und Hypothese h=sign(g) definieren wir den **Margin von** x durch $f(x) \cdot g(x)$.

L liefere Hypothesen $h_1, \ldots, h_T$ mit Fehlern $\varepsilon_1, \ldots, \varepsilon_T$. AdaBoost bestimme die Hypothese $h = \text{sign}(\sum_{t=1}^{T} \alpha_t \cdot h_t)$. Dann gilt für jedes $\gamma \in \mathbb{R}$ und für die Gleichverteilung G auf der Beispielsmenge:

$$\text{prob}_G[\, f(x) \cdot \frac{h(x)}{\sum_{t=1}^{T} \alpha_t} \leq \gamma \,] \leq 2^T \cdot \Pi_{t=1}^{T} \sqrt{\varepsilon_t^{1-\gamma}(1-\varepsilon_t)^{1+\gamma}}.$$

- Der Beweis ist sehr ähnlich zur Analyse des Trainingsfehlers.
- Solange $0 \leq \gamma < \theta \leq \frac{1}{2}$ ist $2^T \cdot \Pi_{t=1}^{T} \sqrt{\varepsilon_t^{1-\gamma}(1-\varepsilon_i)^{1+\gamma}} \leq \Pi_{t=1}^{T} \left(\sqrt{(1-2\cdot\theta)^{1-\gamma}(1+2\cdot\theta)^{1+\gamma}} \right) < 1$.
- „Hochwahrscheinlich“ Margin höchstens $\gamma \cdot (\sum_{t=1}^{T} \alpha_t)$.