Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

INTRUSIONSERKENNUNG MIT HIDDEN MARKOV MODELS

Fachsemimar WS 2015/16

Letztes Update: 27. Januar 2016

Thorsten Knoll

Studienbereich Informatik
Hochschule RheinMain

INHALT

DAS PROBLEM

BESCHREIBUNG DES PROBLEMS

- Intrusionen in Computersysteme
- Beispiele: Trojaner, SymLink Attacken, Buffer Overflows
- Erkennung von Intrusionen in laufenden Systemen mit System Calls

Normal	**Intrusion**
open	open
	getpid
	fork
time	time
read	read
write	write
close	close

Tabelle: System Call Sequenzen

HIDDEN MARKOV MODELS (HMMS)

ANDREI ANDREJEWITSCH MARKOV, MATHEMATIKER, 1856 - 1922

Quelle: Wikipedia, Public Domain (Gemeinfrei)

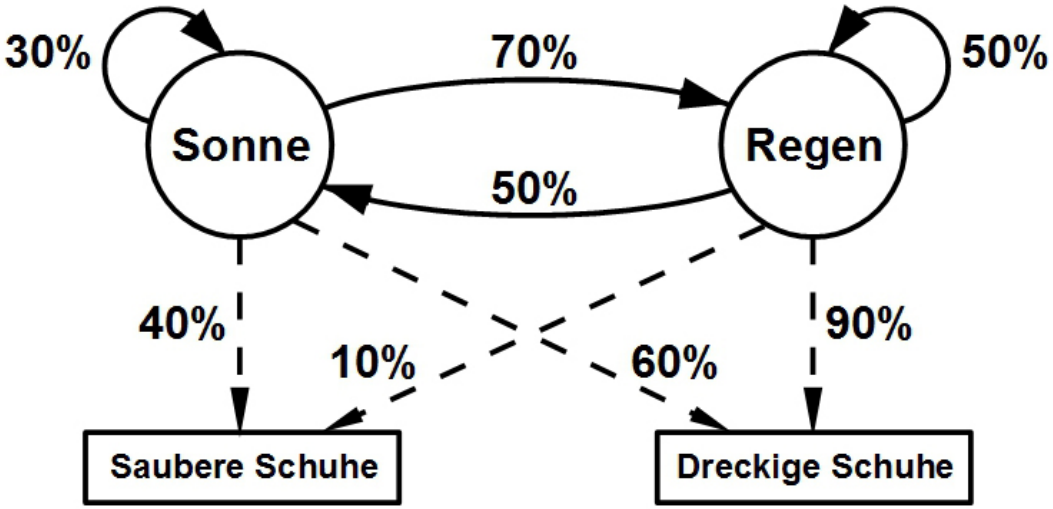
ANWENDUNGSGEBIETE

Anwendungsgebiete:

- Spracherkennung
- Texterkennung
- Genanalyse
- Intrusionserkennung?

KONKRETE BEISPIELE

Der Gefangene im Kerker ohne Fenster:



Quelle: Wikipedia Hidden Markov Models

Spracherkennung:

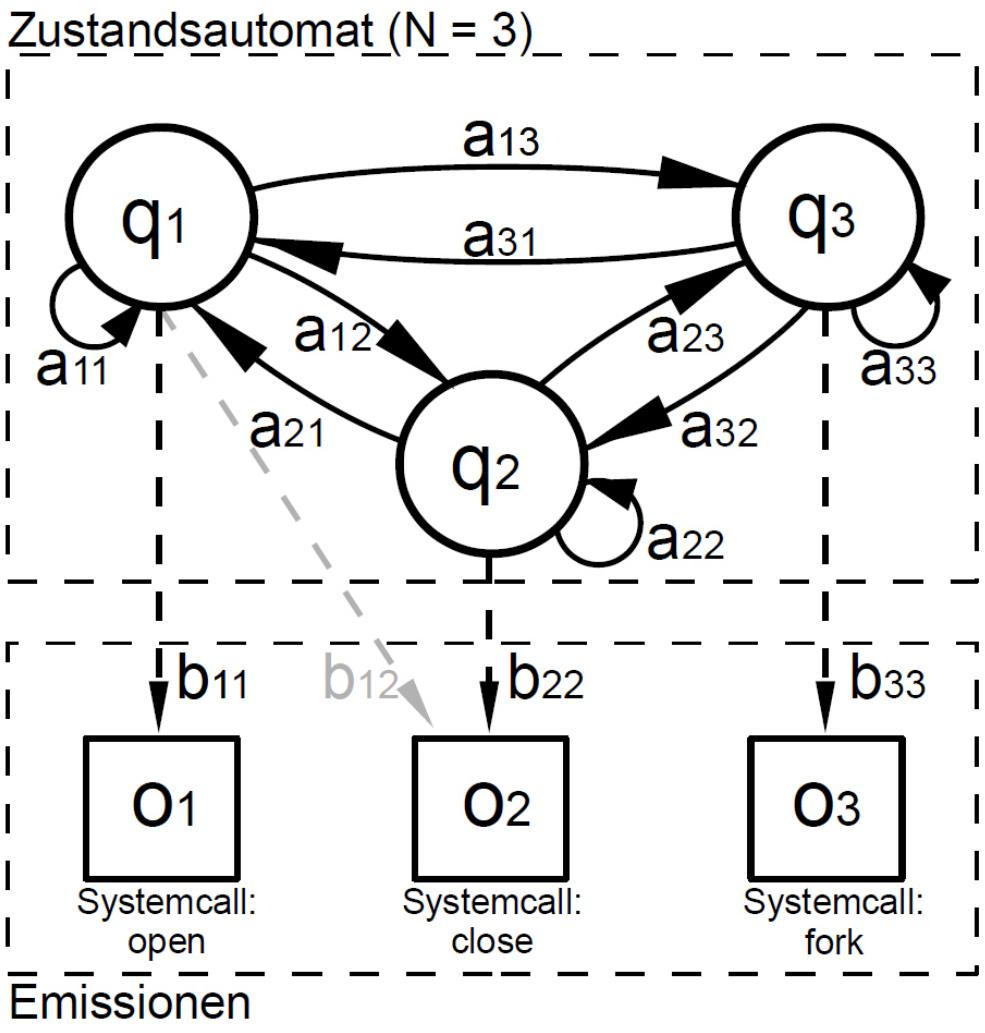
- Versteckte Zustände: gesprochene Laute
- Emissionen: hörbare Töne

ANNAHMEN FÜR HMMS

Annahmen für HMMs:

- Die Anzahl der Zustände ist endlich (N).
- Das Gedächtnis des Automaten ist genau einen Zustand groß (Markov-Eigenschaft).
- Die Zustandsübergange haben Wahrscheinlichkeiten.
- Die Emissionen sind mit Wahrscheinlichkeiten an die Zustände gekoppelt.

DEFINITION VON HMMS



HMM – Definition:

T	Anzahl der Zeitpunkte
N	Anzahl der Zustände
M	Anzahl der mögl. Emissionen
Q	Zustandsmenge {q1,...,qN}
V	Menge der mögl. Emissionsfolgen
O	Emissionsfolgen
A	Matrix der Übergangs-wahrscheinlichk.
B	Matrix der Emissions-wahrscheinlichk.
π	Vektor der Startzustands-wahrscheinlichk.

Abbildung: Beispiel eines HMMs mit 3 Zuständen und Definitionstabelle

KURZSCHREIBWEISE UND MARKOV-EIGENSCHAFT

Kurzschreibweise von HMMs als 3-Tupel:

$$\lambda = (A, B, \pi)$$

KLASSISCHE FRAGESTELLUNGEN ÜBER HHMS

1. Evaluation:

Gegeben: Mehrere HMMs $\lambda_1, ..., \lambda_n$, Observationsfolge O.
Gesucht: Das am besten passendste HMM λ_i zu O.

- Die Produktionswahrscheinlichkeit $P(O \mid \lambda_i)$ berechnen $(1 \leq i \leq n)$.
- Durch Bayes und Maximierung das passendste HMM finden:
- $\lambda_i = \text{argmax}_{\lambda_j}(P(O \mid \lambda_j)P(\lambda_j))$ mit $1 \leq j \leq n$

KLASSISCHE FRAGESTELLUNGEN ÜBER HHMS

2. Decoding:
Gegeben: Ein HMM λ, Observationsfolge O.
Gesucht: Die optimale Zustandsfolge z^* für O.

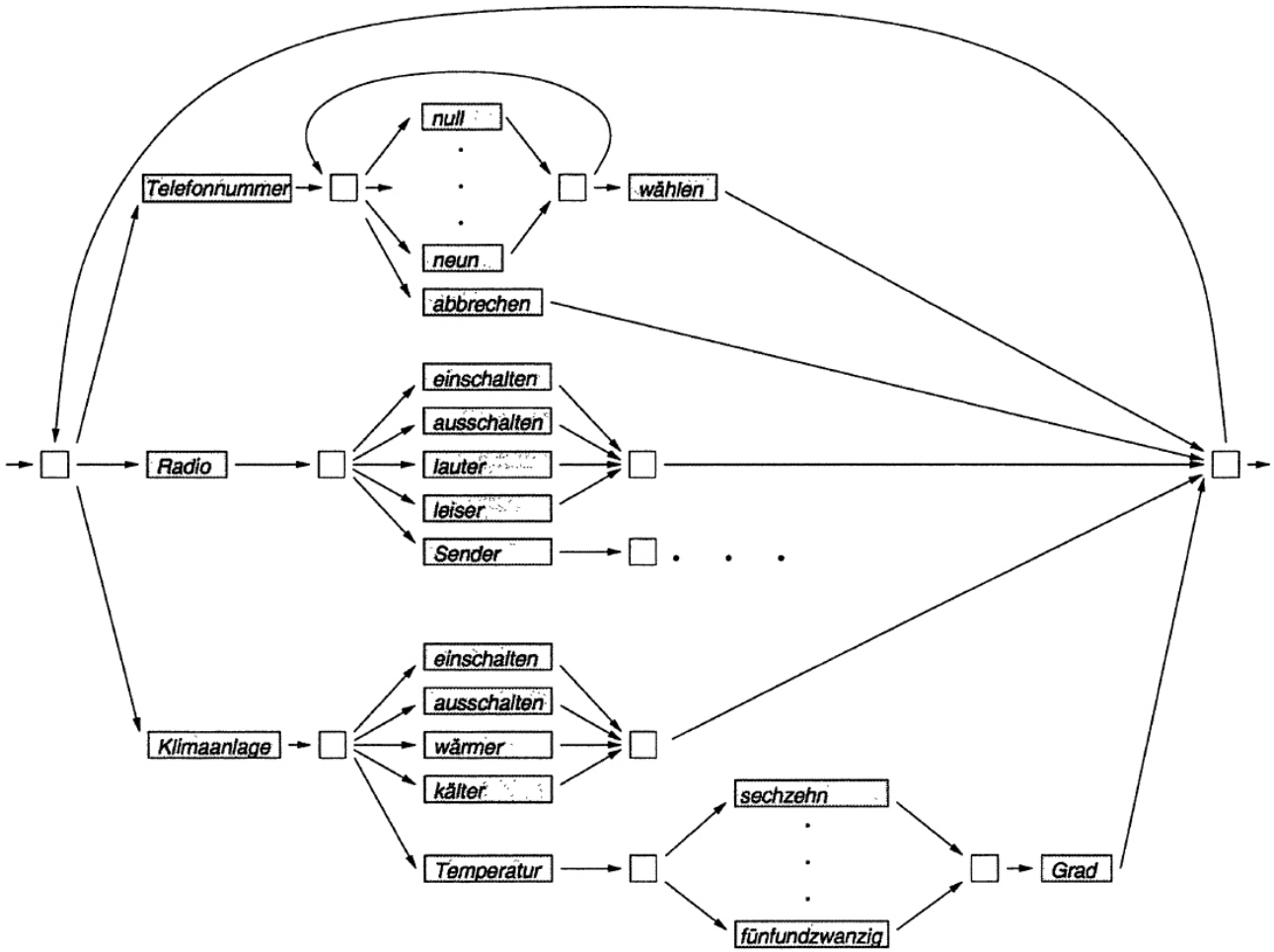
- $z^* = \text{argmax}_z(P(z \mid O, \lambda))$
- Lösung: Viterbi Algorithmus
- Viterbi verwendet den Forward-Backward-Algorithmus

KLASSISCHE FRAGESTELLUNGEN ÜBER HHMS

3. Training, Machine Learning:
Gegeben: Ein HMM $\lambda = (A, B, \pi)$, Trainingsdaten
Gesucht: Parameteroptimierung für A, B, π

- Lösung: Baum-Welch-Algorithmus
- Iteratives Optimieren mit den Trainingsdaten
- Abbruchbedingung: Keine weitere Veränderung der Parameter

VERBUNDMODELLE VON HMMS



Quelle: G.A.Fink, Mustererkennung mit Markov-Modellen, Seite 132

Abbildung: Schematischer HMM-Verbund für Spracherkennung im Auto

ZUSAMMENFASSUNG HMMS

Wir wissen jetzt:

- Wie HMMs definiert sind: $\lambda = (A, B, \pi)$.
- Wie HMMs evaluiert, dekodiert und trainiert werden.
- Wie ein Verbund aus HMMs funktioniert.

Jetzt betrachten wir das Problem der Intrusionserkennung.

INTRUSIONSERKENNUNG

PAPER: DETECTING INTRUSIONS USING SYSTEM CALLS

Die jetzt vorgestellen Daten und Ergebnisse sind aus dem Paper
Detecting Intrusions using System Calls
von **Warrender, Forrest, Pearmutter**
an der **University New Mexico**

TRAININGS- UND TESTDATEN

Program	Intrusions	Normal data available		Normal data used for training		Normal data used for testing	
	Number of traces	Number of traces	Number of system calls	Number of traces	Number of system calls	Number of traces	Number of system calls
MIT `lpr`	1001	2,703	2,926,304	415	568,733	1,645	1,553,768
UNM `lpr`	1001	4,298	2,027,468	390	329,154	2,823	1,325,670
`named`	2	27	9,230,572	8	677,340	12	7,690,572
`xlock`	2	72	16,937,816	72	778,661	1	16,000,000
`login`	9	12	8,894	12	8,894	–	–
`ps`	26	24	6,144	24	6,144	–	–
`inetd`	31	3	541	3	541	–	–
`stide`	105	13,726	15,618,237	150	246,750	13,526	15,185,927
`sendmail`	–	71,760	44,500,219	4,190	2,309,419	57,775	35,578,249

Quelle: Warrender, Forrest, Pearlmutter, Detecting Intrusions using System Calls, Seite 4

Abbildung: Test- und Trainingsdaten für die Intrusionserkennung

Die Daten sind verfügbar unter:
http://www.cs.unm.edu/~immsec/data-sets.html

ERKENNUNGSMETHODEN

Zum Vergleich verwendete Erkennungsmethoden:

- Enumerationsverfahren: stide, t-stide
- Data Mining Verfahren: RIPPER
- Hidden Markov Models

MODELLIERUNG, TRAINING UND TESTEN DER HMMS

Modellierung:

- Ein HMM für jedes Programm
- Größe der Zustandsautomaten: jeweils 20-60 Zustände
- Anfangs-Wahrscheinlichkeiten wurden zufällig gewählt

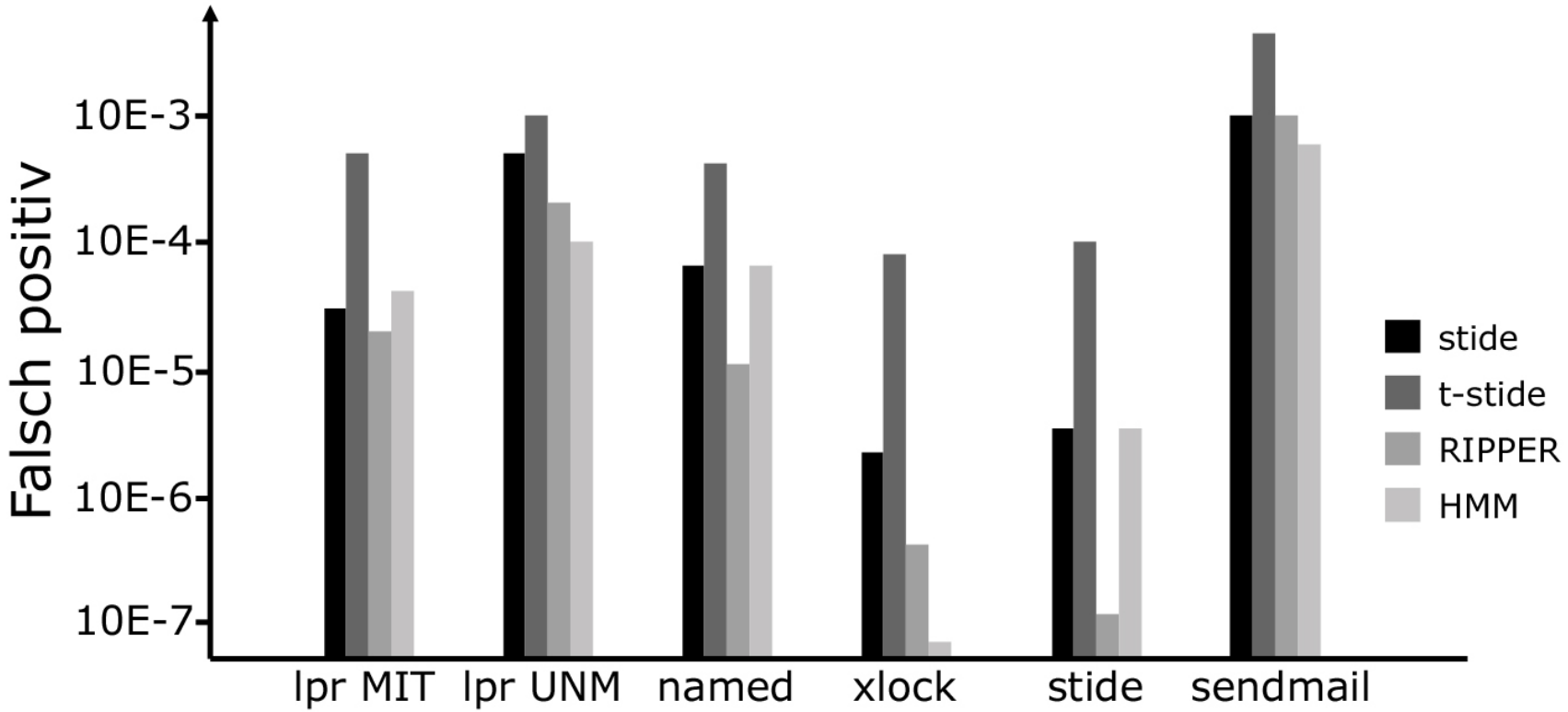
Training:

- Mit den Normal-Daten (ohne Intrusionen)
- Trainingsdauer für die HMMs bis zu mehrere Monate!

Testen:

- Mit der Produktionswahrscheinlichkeit in positiv / negativ einteilen
- Daraus ergeben sich die Ergebniskategorien:
- falsch / richtig positiv
- falsch / richtig negativ

ERGEBNISSE DES VERGLEICHS



Quelle: Warrender, Forrest, Pearlmutter, Detecting Intrusions using System Calls, Seite 11

Abbildung: Ergebnisse des Vergleichs

FAZIT, QUELLEN

FAZIT

Fazit:
HMMs sind gut für die Intrusionserkennung...

- aber nicht besser als die weniger aufwändige Enumeration (stide)
- und brauchen sehr lange für das Training

QUELLEN

- Shaunak Chatterjee and Stuart Russell, *A temporally abstracted viterbi algorithm*, Uncertainty in Artificial Intelligence (UAI), AUAI Press, 2011, pp. 96–104.
- Gernot A. Fink, *Mustererkennung mit markov-modellen : Theorie, praxis, anwendungsgebiete*, Ph.D. thesis, Stuttgart [u.a.], 2003, Zugl.:Bielefeld, Univ., Habil.-Schr., 2003.
- L. Rabiner and B.H. Juang, *An introduction to hidden markov models*, ASSP Magazine, IEEE **3** (1986), no. 1, 4–16.
- C. Warrender, S. Forrest, and B. Pearlmutter, *Detecting intrusions using system calls: alternative data models*, Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on, 1999, pp. 133–145.