Exponentialfamilien auf Ressourcenbeschränkten Systemen[1]

Nico Piatkowski[2]

Abstract: Um Maschinelles Lernen (ML) in sicherheitskritischen oder autonomen Systemen einzusetzen sind Gütegarantien und Fehlerschranken erforderlich—eine rein empirische Evaluation von gelernten Modellen reicht für solche Systeme nicht aus. Insbesondere für Methoden die heuristisch motiviert sind, ist eine Herleitung solcher Garantien oft nicht möglich. Dies gilt in der Tat nicht für alle ML Verfahren: Die Exponentialfamilie ist eine Klasse probabilistischer Modelle die es uns erlaubt, das Wahrscheinlichkeitsmaß diskreter Daten zu lernen ohne dabei spezielle Annahmen über die zugrundeliegende Verteilung zu machen. Mitglieder der Exponentialfamilie bilden die Grundlage für eine Vielzahl der heute gängigen Techniken des Maschinellen Lernens, darunter Logistische Regression, Markov Random Fields und Boltzmann Maschinen. Im Gegensatz zu Ansätzen des „Deep-Learning" erlauben generative probabilistische Modelle die konsistente Schätzung der gemeinsamen Verteilung aller beteiligten Zufallsvariablen, sowie der inhärenten bedingten Unabhängigkeitsstruktur. Einmal "gelernt", ermöglichen uns solche Modelle sowohl die Klassifikation von Ereignissen, als auch das Sampling neuer Daten. Diese herausragenden theoretischen Eigenschaften werden jedoch von einer hohen Worst-Case-Komplexität begleitet. Um Exponentialfamilienmodelle in der Praxis—außerhalb von Großrechnern—einzusetzen, ist es von essentieller Bedeutung zu verstehen, unter welchen Umständen der Worst-Case tatsächlich eintritt und wie die Komplexität mit Hilfe von Approximationsalgorithmen verringert werden kann. Daher wird in dieser Arbeit die Klasse der Exponentialfamilien systematisch bezüglich ihres Ressourcenbedarfs untersucht. Aus der formalen Spezifikation der Modellklasse werden Approximationen hergeleitet, die den Speicherverbrauch, die erforderlichen arithmetischen Instruktionen sowie die Berechnungskomplexität der gesamten Modellklasse reduzieren. Dabei wird stets sichergestellt, dass der zusätzliche Fehler, der durch unsere Approximationen entsteht, beschränkt ist. Die theoretischen Erkenntnisse werden auf Datensätzen realer Anwendungen validiert.

[1] Englischer Originaltitel: "Exponential Families on Resource-Constrained Systems"

[2] TU Dortmund, Lehrstuhl für Künstliche Intelligenz, nico.piatkowski@tu-dortmund.de

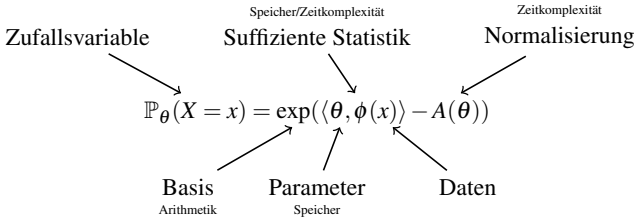

Abb. 1: Wesentliche Bestandteile eines Exponentialfamilienmodells für eine (n-dimensionale) Zufallsvariable X sowie die hauptsächlich verbrauchte Ressource.

1 Einleitung

Die Künstliche Intelligenz und insbesondere das Maschinelle Lernen erfahren zur Zeit große Aufmerksamkeit in Medien und Politik. Doch auch wenn geringe Klassifikationsfehler mit teilweise übermenschlicher Performanz erreicht werden, reicht eine rein empirische Evaluation für viele Anwendungen nicht aus: Für den Einsatz von ML in sicherheitskritischen oder autonomen Systemen sind Gütegarantien und Fehlerschranken erforderlich. Beispielhafte Szenarien sind autonome Kraftfahrzeuge, Drohnen, oder mobile medizinische Geräte.

Ferner ist der Ressourcenverbrauch von state-of-the-art ML Methoden oft auf Großrechner ausgelegt: Auch wenn es zuerst beeindruckend klingt, dass IBM's Watson „Jeopardy!"-Sieger geworden ist oder das Google DeepMind's AlphaGo einen der weltbesten Spieler des Brettspiels Go besiegen konnte, so sind diese Erfolge weniger beeindruckend wenn man den Ressourcenverbrauch betrachtet. Watson bestand zum Zeitpunkt des Sieges aus neunzig Servern und AlphaGo bestand aus einem System mit 1920 CPUs sowie 280 GPUs. Der genaue Energieverbrauch beider Systeme ist unbekannt, aber die verwendete Hardware legt nahe, dass ihre Leistungsaufnahme bei ca. einhunderttausend Watt liegt—die durchschnittliche Leistungsaufnahme des menschlichen Gehirns liegt bei ca. 20 Watt.

Beide Punkte sind für diese Arbeit von zentraler Bedeutung: Zum einen müssen wir Garantien über das Lernverhalten abgeben können, beispielsweise in Form von asymptotischer Konsistenz oder einer beschränkten Abweichung vom optimalen Lernergebnis. Zum anderen müssen wir den Ressourcenverbrauch von Lernverfahren verstehen und gegebenenfalls an die verfügbare Hardware anpassen können.

Die Menge *aller* Lernverfahren ist selbstverständlich zu groß und zu unstrukturiert für eine systematische Analyse. Wir beschränken uns hier auf die Klasse der Exponentialfamilien, um sowohl die theoretischen Eigenschaften als auch den Ressourcenverbrauch systematisch zu untersuchen. Die funktionale Form sowie die wesentlichen Bestandteile von Verteilungen der Exponentialfamilie sind in Abbildung 1 dargestellt. Die kanonische Verlustfunktion von Exponentialfamilien ist die *negative log-Likelihood* $-\ell(\theta) = -\sum_{x \in \mathscr{D}} \log \mathbb{P}_\theta(x)$. Beim "Lernen" wird diese Funktion bezüglich der Parameter $\theta \in \mathbb{R}^d$ mittels numerischer Optimierungsverfahren minimiert.

Die Klasse der Exponentialfamilienmodelle enthält wichtige Modelle des Maschinellen Lernens, darunter Markov Random Fields, Conditional Random Fields, Lineare- und Logistische Regression und Boltzmann-Maschinen, sowie viele bekannte Wahrscheinlichkeitsverteilungen wie die Normalverteilung, die Poisson-, die Dirichlet- und die Kategorische-Verteilung. Jeder Fortschritt der auf dem Gebiet der Exponentialfamilien erzielt wird, wirkt sich also direkt auf eine Vielzahl von theoretischen Modellen und praktischen Anwendungen aus.

Im Folgenden widmen wir uns drei zentralen Fragestellungen, deren Untersuchung uns letztendlich erlauben wird, Exponentialfamilienmodelle selbst auf sehr schwachen (sog. "ultra-low-power") Berechnungsarchitekturen mit beschränktem Approximationsfehler zu lernen und anzuwenden. Wir beschränken uns hier jeweils auf die Kernresultate in Form von Theoremen sowie einige experimentelle Resultate. Die zugehörigen Beweise sowie vollständige Beschreibung des experimentellen Designs können in [Pi18] gefunden werden. Alle der im Folgenden gezeigten Experimente wurden auf den folgenden drei Datensätzen durchgeführt:

- INSIGHT—SCATS Daten der Stadt Dublin: Der Datensatz enthält Messungen von 2367 SCATS (Sydney Coordinated Adaptive Traffic System) Verkehrs Sensoren, welche in diversen Straßen der Stadt Dublin eingebettet sind. Die Daten wurden zwischen dem 1. Januar 2013 und dem 14. Mai 2013 erhoben. Jeder Sensor gibt pro Minute die Durchschnittsgeschwindigkeit sowie die relative Sensorbelegung (Verkehrsdichte) aus.
- VaVeL—Mobilfunknetzwerknutzung der Stadt Warschau: Der Datensatz enthält die Auslastung von 4988 Zellen des Mobilfunknetzes der Stadt Warschau. Die Daten wurden zwischen dem 15. Mai 2016 und dem 26. Juni 2016 erhoben. Jeder Datenpunkt enthält Ereignisse über An- und Abmeldungen der Nutzer sowie Nutzung von Sprach- und xMS Kanälen. Die Daten jeder Mobilfunkzelle wurden stundenweise aggregiert.
- Intel Lab—Temperatur und Luftfeuchtigkeit: Der Datensatz enthält Messungen der Temperatur und Luftfeuchtigkeit von 56 Sensoren im Intel Berkeley Research Lab. Die Daten wurden zwischen dem 28. Februar und dem 2. April 2004 erhoben. Jeder Sensor erzeugt alle 31 Sekunden eine neue Messung.

Die Daten jedes Datensatzes wurden in gleichlange Sequenzen über 24 Stunden partitioniert. Für jeden Datensatz wurde die bedingte Unabhängigkeitsstruktur, welche die stochastische Interaktion zwischen den Sensoren repräsentiert, mit dem Chow-Liu-Algorithmus [CL68] bestimmt.

2 Modelkompression durch Regularisierte Reparametrisierung

Im ersten Beitrag geht es um das Lernen komprimierter Modelle und die damit verbundene Reduktion des Speicherverbrauchs. Ist die zugrundeliegende Zufallsvariable hochdimensional, so müssen oft mehrere Millionen Modellparameter gelernt werden. In Abhängigkeit

von der gelernten Struktur kann die Anzahl der Parameter für die VaVeL Daten 310×10^6 übersteigen. Probabilistische Modelle erlauben eine Interpretation der gelernten Gewichte als bedingte Transinformation zwischen Zufallsvariablen. Bei genauerer Betrachtung fällt auf, dass viele der Parameter redundant sind. Motiviert durch diese Beobachtung wurde die folgende Reparametrisierung untersucht [PLM13]:

$$\theta(\Delta) = D\Delta \tag{1}$$

Hierbei ist θ der klassische Parametervektor, Δ ist der neue Vektor, und D ist eine unitrianguläre untere Dreiecksmatrix mit Einträgen in $[0;1]$. Intuitiv kann die Matrix D so gewählt werden, dass jede Komponente von θ eine Linearkombination von Komponenten von Δ sind. Anstatt θ wird nun Δ mittels numerischer Minimierung von $-\ell(\Delta) = -\sum_{x\in\mathscr{D}} \log \mathbb{P}_{\theta(\Delta)}(x) + \lambda_1 \|\Delta\|_1 + \lambda_2 \|\Delta\|_2^2$ gelernt. Durch die normbasierte *Regularisierung* von Δ ist sichergestellt, dass Redundanzen in θ mittels einiger weniger Dimensionen von Δ abgedeckt werden—der Vektor Δ ist spärlich besetzt wann immer θ viele Redundanzen enthält. Für (1) haben wir gezeigt, jedes θ verlustfrei dargestellt werden kann. Die Reparametrisierung (1) ist also *universell* und kann auch eingesetzt werden, falls das optimale Modell gar keine Redundanzen enthält. Zusätzlich haben wir gezeigt, dass bei korrekter Wahl von λ_1 und λ_2 der Abstand des gelernten Modells zum optimalen Modell stets beschränkt ist:

Theorem 2.1 (Beschränkter Fehler) *Sei X eine Zufallsvariable mit Exponentialfamiliendichte und Parameter $\theta^* \in \mathbb{R}^d$, dessen Reparametrisierung minimale Norm besitzt. Sei $\mathscr{D}$ ein Datensatz mit $N = |\mathscr{D}|$ Realisierungen von X. Nimm an, dass $\|\nabla^2 A(\theta^*)^{-1}\|_\infty \leq \kappa$ und $\|\Delta\|_\infty \leq \gamma$, setze $\lambda_1 = 4T\sqrt{\log(d)/N}$ und $\lambda_2 = \gamma^{-1}\lambda_1$. Wenn $N \geq 324\kappa^4 d^{12}\log(d)/(T - d^2)^2$, dann gilt für jede Wahl von D:*

(I) Der Abstand zwischen dem optimalen Modellparameter θ^* und dem gelernten Schätzer $\eta_D(\hat{\Delta})$ ist beschränkt: $\|\eta_D(\hat{\Delta}) - \theta^*\|_\infty \leq 3\kappa d^2 \lambda_1$,

(II) jede Spärlichkeit im gelernten Modell impliziert eine Redundanz im optimalen Modellparameter: $\hat{\Delta}_{C=x'}(t) = 0 \Rightarrow$

$$|\theta^*_{C=x'}(t-1) - \theta^*_{C=x'}(t)| \quad \leq \quad \frac{3d^2\kappa\lambda_1}{T} + (t-1)\left(\max_{i=1}^{t-1} |\hat{\Delta}_{C=x'}(i)| + \frac{3d^2\kappa\lambda_1}{T}\right),$$

für jede Clique C der bedingten Unabhängigkeitsstruktur und jeden Zeitpunkt t. Beide Aussagen gelten mit einer Wahrscheinlichkeit von mindestens $1 - (2/d)$.

Experimentelle Ergebnisse sind in Abb. 2 dargestellt. Dort bezeichnet M1 das Resultat einer gewöhnlichen Exponentialfamilienschätzung, M2 bezeichnet das Resultat einer l_1-regularisierten Exponentialfamilie und alle anderen Resultate entsprechen der hier vorgestellten regularisierten Reparametrisierung mit verschiedenen *Zerfallsmatritzen D*. Jeder Punkt ist einen Mittelwert—insgesamt wurden 32805 einzelne Experimente durchgeführt. Die Kompression wird hier durch die Parameterdichte $\rho = (\sum_{i=1}^d 1(\Delta_i \neq 0))/d$ gemessen,

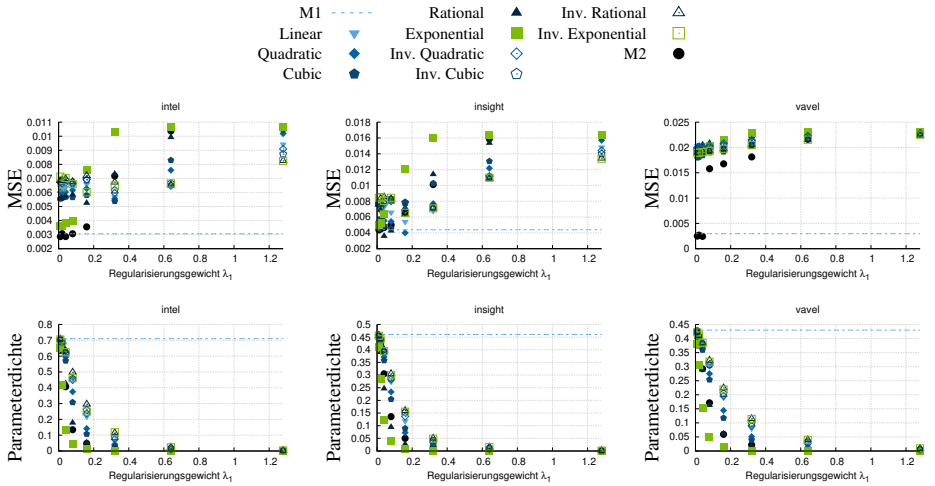

Abb. 2: Experimentelle Ergebnisse der regularisierten Reparametrisierung. Zu sehen ist der mittlere quadratische Fehler (MSE) der geschätzten Randwahrscheinlichkeiten sowie die Dichte der Reparametrisierung Δ als Funktion von λ_1. Verschiedene Farben indizieren verschiedene Wahlen für D. Schwarze Kreise zeigen Resultate für das Modell ohne Reparametrisierung.

also die relative Anzahl der Parameter die nicht Null sind. Je kleiner dieser Wert, umso größer die Kompression des Modells. Je größer der Wert von λ_1, desto stärker versucht das Lernen ein möglichst kleines Modell zu finden. Wir sehen, dass für Kompressionsstärken zwischen 0.2 und 0.8 der Fehler unserer reparametrisierten Modelle vergleichbar oder geringer als der Fehler des M2-Modells ist, während eine ähnliche oder höhere Kompression erreicht wird. Die Ergebnisse zeigen auch, dass das Model der VaVel Daten am wenigsten Redundanzen enthält, da die Kompression hier zu einem größeren Fehler führt.

3 Lernen und Inferenz ohne Gleitpunktarithmetik

Im zweiten Beitrag geht es um das Lernen ganzzahliger Modelle und die damit verbundene Reduktion der erforderlichen Schaltwerkskomplexität. Kleine Geräte oder Sensoren sind oft nur mit schwachen Mikroprozessoren ausgestattet, um einen möglichst geringen Energieverbrauch zu garantieren. Aus diesem Grund wird oft auf zusätzliche Hardware wie Gleitkommakoprozessoren verzichtet. Aus der Modellspezifikation (Abb. 1) ist jedoch klar, dass die Auswertung von Exponentialfamilienmodellen, und damit auch das Lernen und die Anwendung des Modells, inhärent auf der Auswertung transzendenter Funktionen basiert. Bei genauerer Betrachtung der Herleitung der Exponentialfamilie [Pi36] fällt auf, dass diese äquivalent zu jeder anderen Basis hergeleitet werden kann:

Theorem 3.1 (Basis-b Familien) *Jedes Mitglied der Exponentialfamilie kann in Basis-b-Form gebracht werden, d.h.,* $\mathbb{P}_{b,\theta}(X = x) = b^{\langle\theta,\phi(x)\rangle - A_b(\theta)}$ *mit* $A_b(\theta) = \log_b Z_b(\theta) =$

$\log_b \int_{\mathscr{X}} b^{\langle \theta, \phi(x) \rangle} \, \mathrm{d}\nu(x)$. *Ein Wechsel der Basis ist mittels der Reparametrisierung* $\eta_{a,b}(\theta) = (\log b / \log a)\theta$ *möglich:* $\forall x \in \mathscr{X} : \mathbb{P}_{a,\theta}(x) = \mathbb{P}_{b,\eta_{a,b}(\theta)}(x)$.

Durch die Wahl von $b = 2$ sowie der Nebenbedingung $\theta \in \mathbb{N}^d$ erhalten wir auf natürliche Weise eine Teilklasse der Exponentialfamilien, für welche der Abstand zur Klasse aller Exponentialfamilienmodelle beschränkt ist:

Theorem 3.2 (Log-Likelihood Fehler) *Sei* $\lfloor \theta \rfloor$ *das elementweise Abrunden von* $\theta \in \mathbb{R}^d$ *und sei ferner* $\varepsilon = \theta - \lfloor \theta \rfloor$. *Nimm an, dass* θ *der Parameter einer Exponentialfamilie mit übervollständiger suffizienter Statistik ist. Dann gilt* $\ell(\lfloor \theta \rfloor; \mathscr{D}) - \ell(\theta; \mathscr{D}) \leq 2\|\varepsilon\|_2 |\mathscr{C}(G)|$, *wobei Gleichheit erreicht wird wenn* θ *bereits ganzzahlig war.*

Hierbei bezeichnet $|\mathscr{C}(G)|$ die Anzahl der Cliquen in der bedingten Unabhängigkeitsstruktur. Durch die Kombination von Basis-2 Familien mit ganzzahligen Parametern kann die transzendente exp-Funktion außerdem durch einen einfachen Bit-shift ersetzt werden:

Lemma 3.1 (Bit-Shift Potentialfunktion) *Die Potentialfunktion* $\psi_2(x) = 2^{\langle \theta, \phi(x) \rangle}$ *der Basis-*2 *Familie kann via Bit-shift "von links" (*$a \ll b$ *für* $a, b \in \mathbb{N}$*) ausgewertet werden:* $\psi_2(x) = 1 \ll \langle \theta, \phi(x) \rangle$. *Reellwertige Arithmetik ist nicht erforderlich.*

Die Berechnung von Wahrscheinlichkeiten in Exponentialfamilienmodellen erfolgt mit Hilfe von Inferenzalgorithmen. Für die Basis-2-Familie haben wir einen speziellen Inferenzalgorithmus entwickelt, der die Wahrscheinlichkeiten allein mittels ganzzahliger Arithmetik approximieren kann. Kern des Algorithmus ist eine Datenstruktur für dünnbesetzte ganzzahlige Daten sowie die Beobachtung, dass der Logarithmus zur Basis-2, dessen Auswertung für die Inferenz erforderlich ist, durch die Bitlänge einer ganzen Zahl approximiert werden kann. Für die Methode haben wir gezeigt, dass der Approximationsfehler durch Eigenschaften der zugrundeliegenden bedingten Unabhängigkeitsstruktur beschränkt ist, insbesondere durch die Länge des längsten Pfades im Modell. Beispielhafte experimentelle Ergebnisse sind in Abb. 3 dargestellt. Dort vergleichen wir exakte und approximative-ganzzahlige Inferenz auf zwei synthetischen bedingten Unabhängigkeitsstrukturen. Zum einen „chain", welche einer linearen Liste entspricht, und „star", ein sternförmiger Graph. Die Länge des längsten Pfades in "star" ist konstant 2 (unabhängig von der Anzahl der Variablen), während die längste Pfad in "chain" gleich der Anzahl an Variablen ist. Die Modellparameter wurden zufällig Normalverteilt mit Mittelwert 0 und verschiedenen Varianzen erzeugt und abschließend zum nächst kleineren ganzzahligen Wert gerundet. Die Ergebnisse spiegeln zum einen unsere theoretische Einsicht wieder, dass der Fehler hauptsächlich durch die Länge des längsten Pfades beeinflusst wird. Zum anderen sehen wir, dass die Einschränkung auf ganzzahlige Modellparameter den Raum der Wahrscheinlichkeiten diskretisiert: das Modell kann nur eine kleine endliche Teilmenge an verschiedenen Wahrscheinlichkeiten erzeugen.

Da wir nun in der Lage sind, Inferenz in ganzzahligen Modellen mit beschränktem Fehler zu betreiben, ist der letzte Schritt die Herleitung einer Methode für das Lernen ganzzahliger Modellparameter. Dazu führen wir eine neue Regularisierung ein:

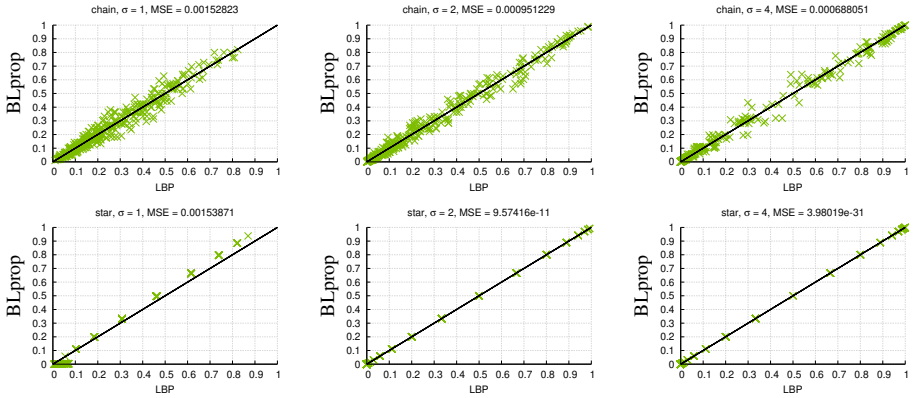

Abb. 3: Vergleich der berechneten Randwahrscheinlichkeiten von exakter Inferenz (LBP) sowie der approximativen ganzzahligen Inferenz (BLprop) auf zwei bedingten Unabhängigkeitsstrukturen („chain" und „star") für zufällige Parameter mit unterschiedlicher Standardabweichung $\sigma \in \{1,2,4\}$.

Definition 3.1 (Integer-Regularisierung) *Die Integer-Regularisierung bestraft den Abstand jedes Modellparameters zum nächsten ganzzahligen Wert:*

$$R_{int}(\theta) = \sum_{i=1}^{d} \rho_{int}(\theta_i) \qquad mit \qquad \rho_{int}(\theta_i) = 1 - |1 - 2(\lceil \theta_i \rceil - \theta_i)| \,.$$

Das Modell wird also durch die numerische Minimierung von $-\ell(\theta) = -\sum_{x \in \mathscr{D}} \log \mathbb{P}_\theta(x) + \lambda R_{\mathtt{int}}(\theta)$ gelernt. Da diese Regularisierung sowohl nicht-stetig als auch nicht-konvex ist, musste ein spezieller proximaler-Gradientenabstieg [PB14] inklusive Konvergenzbeweis hergeleitet werden. Entsprechende Ergebnisse sind in Abb. 4 dargestellt. Hier wurde zusätzlich die maximale Bitlänge (b) der gelernten ganzzahligen Parameter eingeschränkt, um zu zeigen, dass keine (triviale) Fixpunktdarstellung der reellwertigen Parameter gelernt wird. Tatsächlich zeigen die Ergebnisse, dass bereits drei Bits ausreichen, um Modelle mit geringem Approximationsfehler zu lernen, während die Laufzeit um ein Vielfaches geringer ist. Hierbei ist zu beachten, dass die Laufzeit auf einer gewöhnlichen Workstation gemessen wurde. Im letzten Abschnitt gehen wir kurz auf Ergebnisse ein, die auf einem Mikrocontroller erzielt werden konnten. Zusätzliche Ergebnisse zu Klassifikationsexperimenten können in [PLM16] gefunden werden.

4 Lernen und Inferenz durch Stochastische Quadratur

Im dritten Beitrag befassen wir uns mit der Laufzeitkomplexität der probabilistischen Inferenz. Dies entspricht der Auswertung der Normalisierungskonstante $A(\theta) = \log Z(\theta)$ aus Abb. 1. Es kann gezeigt werden, dass dieses Problem #P-vollständig ist [Va79]. Hierbei ist es wichtig zu verstehen, dass diese Eingruppierung der Komplexität den Worst-Case über die Wahl der bedingten Unabhängigkeitsstruktur widerspiegelt. Aus diesem Grund

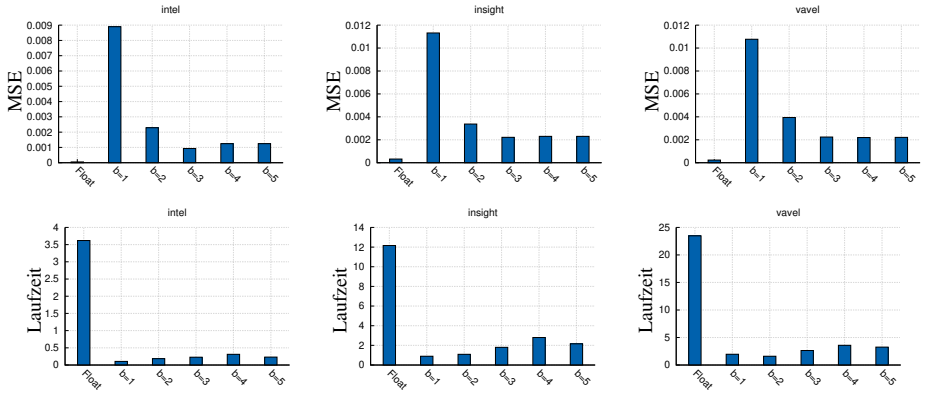

Abb. 4: Experimentelle Ergebnisse der ganzzahligen Exponentialfamilienmodelle. Zu sehen ist der mittlere quadratische Fehler (MSE) der geschätzten Randwahrscheinlichkeiten sowie die Laufzeit als Funktion der maximalen Bitlänge (Wortbreite) pro Parameter. „Float" bezeichnet das gewöhnliche (nicht ganzzahlige) Exponentialfamilienmodell.

basieren die meisten Approximationsalgorithmen für probabilistische Inferenz auf einer Approximation dieser Struktur und damit auf einer Elimination von Abhängigkeiten, die zwischen Zufallsvariablen existieren. Da die Berücksichtigung aller Abhängigkeiten aber gerade die Eigenschaft ist, die es uns erlaubt, Garantien über das Lernergebnis abzugeben, haben wir versucht, eine andere Art der approximativen Inferenz zu finden, bei der die Struktur intakt bleibt. Unser neuer Ansatz [PM16] basiert auf dem Konzept der Quadratur:

$$I[f] = \int_l^u f(x)\mathrm{d}x \approx \int_l^u h_k(x)\mathrm{d}x = \sum_{i=0}^{k} w_i f(x_i) = I_k[f] \tag{2}$$

Hierbei ist $I[f]$ ein Integral über f, dessen Auswertung aufgrund zu hoher Komplexität nicht möglich ist. Stattdessen wird eine Approximation von f so gewählt, dass das Integral doch berechnet werden kann. War der Approximationsfehler zwischen f und h_k beschränkt, so ist auch der Fehler der Integralapproximation beschränkt. Dies wenden wir nun auf die Normalisierungskonstante $Z(\theta)$ an. Dazu wählen wir als h_k eine Chebychev-Polynominterpolation von $\exp(\langle\theta,\phi(x)\rangle)$ zum Grad k. In diesem Fall spricht man von einer Clenshaw-Curtis Quadratur [CC60]. Da die resultierende Quadraturapproximation von $Z(\theta)$ einem mehrdimensionalen Polynom mit d^k Summanden entspricht, berechnen wir die endgültige Approximation auf einer zufälligen Stichprobe von Summanden—dieses Vorgehen wird auch *Stochastische Quadratur* genannt. Da das Erstellen einer zufälligen Stichprobe möglichst effizient sein muss, haben wir einen spezialisierten Monte-Carlo Stichprobenalgorithmus entwickelt [PM18]. Für die endgültige Prozedur konnten wir folgende Fehlerschranke herleiten:

Theorem 4.1 (Fehlerschanke) *Sei ζ der Koeffizientenvektor einer Grad-k Chebychev-Polynomapproximation von* exp *auf $[l;u] = [-\|\theta\|_1;+\|\theta\|_1]$ mit Worst-Case Fehler ε. Sei $\hat{Z}_\zeta^{N,k}(\theta)$ die Ausgabe des neuen Inferenzalgorithmus'. Sei ferner $\delta \in (0,1]$, $\varepsilon > 0$,*

$N = (\log 2/\delta)\tau^2 2\|\theta\|_\infty^{2k'}\varepsilon^{-2}|\mathscr{X}|^{-2}$, *mit* $(k-1)\,k! \geq 8\exp(2\|\theta\|_1)/(\pi\varepsilon)$, *und* $k' = 1$ *wenn* $\|\theta\|_\infty < 1$ *oder andernfalls* $k' = k$. *Dann gilt*

$$\mathbb{P}[|\hat{Z}_\zeta^{N,k}(\theta) - Z(\theta)| < \varepsilon Z(\theta)] \geq 1 - \delta\ .$$

Eine besondere Einsicht dieses Resultats ist der Zusammenhang zwischen der Norm des Parametervektors $\|\theta\|_1$ und dem Approximationsfehler. Bisherige approximative Inferenzmethoden basieren auf Einschränkungen der Struktur. Unser Theorem zeigt jedoch, dass eine Einschränkung der Parameternorm ausreicht, um den erforderlichen Polynomgrad k zu senken, was wiederum eine Senkung der Worst-Case Komplexität $\mathscr{O}(d^k)$ zur Folge hat. Tatsächlich kann die Norm mittels l_1-Regularisierung $\lambda_1\|\theta\|_1$ während des Lernens kontrolliert werden. Exemplarische Lernkurven sind in Abb. 5 dargestellt. Hierbei ist zu beachten, dass sowohl der Gradient als auch der Zielfunktionswert beim Lernen auf der stochastischen Quadratur basieren. Wir sehen, dass in diesem Fall eine kubische Interpolation bereits ausreicht, um eine sehr gute Näherungslösung zu erhalten.

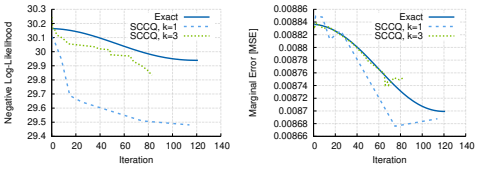

Abb. 5: Exemplarische Lernkurven der Stochastischen Clenshaw-Curtis Quadratur (SCCQ) für eine lineare und eine kubische Approximation.

5 Fazit

Unsere systematische Analyse der Ressourcennutzung von Exponentialfamilienmodellen brachte uns neue Erkenntnisse auf den Gebieten der „Model-Compression", dem Lernen unter nicht-konvexen Nebenbedingungen sowie der approximativen Inferenz unter Erhalt der bedingten Unabhängigkeitsstruktur. Alle unsere Ergebnisse verbindet die Regularisierung, die es uns erlaubt, die Menge der erlaubten Modelle während des Lernens einzuschränken. Durch eine theoretisch fundierte Einschränkung der erlaubten Parameter kann die Worst-Case Abweichung zur unrestringierten optimalen Lösung beschränkt werden. Desweiteren können Regularisierungsterme als log-a-priori Wahrscheinlichkeit interpretiert werden. Daher können unsere Ansätze auch als Bayesianische Schätzung interpretiert werden, wobei die Wahrscheinlichkeit, ein bestimmtes Modell zu lernen, proportional zu seinem Ressourcenverbrauch ist. Abschließend sind in Abb. 6 Laufzeitresultate für unsere ganzzahlige Inferenz (BLprop) sowie unsere stochastische Quadratur (SCCQ) auf einem MSP430 Mikrocontroller mit 16 MHz Takt dargestellt. LBP bezeichnet hier die gewöhnliche (Loopy)-Belief-Propagation. Hier können wir eine $>$ 200-fache Beschleunigung beobachten.

Abb. 6: Laufzeit (in ms) einer Iteration bzw. eines Samples der Algorithmen LBP, BLprop und SCCQ.

Daten	$\lvert E\rvert$	**LBP**	**BLprop**	**SCCQ**
Chain	15	4843.4	19.0	773.8
Star	15	4744.3	19.0	774.0
Grid	24	7713.9	29.5	1081.3
Full	120	40422.6	141.2	4704.2

Nico Piatkowski studierte Informatik und Wirtschaftswissenschaften an der Technischen Universität Dortmund. Nach dem Abschluss seines Studiums forschte er dort am Lehrstuhl für Künstliche Intelligenz. Nico promovierte über Maschinelles Lernen unter Ressourcenbeschränkung im Rahmen des Sonderforschungsbereichs 876. Für seine Forschung erhielt er den Best-Student-Paper-Award der größten europäischen Konferenz für maschinelles Lernen (ECMLPKDD) sowie den Dissertationspreis der TU Dortmund. Nico ist Autor von über 30 begutachteten Fachartikeln und regelmäßig als Mitglied in Programmkomitees aller großen Machine Learning und Data Mining Konferenzen, sowie als Gutachter für Fachzeitschriften. Nico organisierte 2012 das jährliche Treffen der GI-Fachgruppe "Knowledge Discovery, Data Mining und Maschinelles Lernen" und half als "Proceedings Chair" bei der Organisation der ECMLPKDD 2013.

Literaturverzeichnis

[CC60] Clenshaw, C. W.; Curtis, A. R.: A method for numerical integration on an automatic computer. Numerische Mathematik, 2(1):197–205, 1960.

[CL68] Chow, C.; Liu, C.: Approximating discrete probability distributions with dependence trees. IEEE Transactions on Information Theory, 14(3):462–467, May 1968.

[PB14] Parikh, Neal; Boyd, Stephen: Proximal Algorithms. Foundations and Trends in Optimization, 1(3):127–239, 2014.

[Pi36] Pitman, Edwin James George: Sufficient statistics and intrinsic accuracy. Mathematical Proceedings of the Cambridge Philosophical Society, 32:567–579, 1936.

[Pi18] Piatkowski, Nico: Exponential families on resource-constrained systems. Dissertation, Technical University of Dortmund, Germany, 2018.

[PLM13] Piatkowski, Nico; Lee, Sangkyun; Morik, Katharina: Spatio-Temporal Random Fields: Compressible Representation and Distributed Estimation. Machine Learning, 93(1):115–139, 2013.

[PLM16] Piatkowski, Nico; Lee, Sangkyun; Morik, Katharina: Integer undirected graphical models for resource-constrained systems. Neurocomputing, 173, Part 1:9–23, 2016.

[PM16] Piatkowski, Nico; Morik, Katharina: Stochastic Discrete Clenshaw-Curtis Quadrature. In: Proceedings of the ICML. Jgg. 48 in JMLR Workshop and Conference Proceedings. JMLR.org, S. 3000–3009, 2016.

[PM18] Piatkowski, Nico; Morik, Katharina: Fast Stochastic Quadrature for Approximate Maximum-Likelihood Estimation. In: Proceedings of the UAI. 2018.

[Va79] Valiant, Leslie Gabriel: The complexity of enumeration and reliability problems. SIAM Journal on Computing, 8(3):410–421, 1979.