Die Quelle der Sprachfähigkeit in einem energiebasierten assoziativen Substrat

Lokale Regeln, Gradientenabstieg und ein datengewachsener Kontextgraph auf einem gemeinsamen Zeichenebenen-Harness

Marcel Langjahr

Pantareon GmbH
https://pantareon.io/
langjahr@pantareon.io

Juni 2026

Zusammenfassung

Wir stellen eine enge, aber entscheidende Frage: Wenn ein dezentrales, energiebasiertes assoziatives Substrat dazu gebracht wird, Sprache zu modellieren, kommt die Fähigkeit dann aus der nativen Dynamik des Substrats, aus der auf es angewandten Lernregel—oder aus keinem von beiden? Mit einem gemeinsamen Zeichenebenen-Harness—identische Tokenisierung, ein expliziter Train-/Validierungs-Split und eine held-out-Bits-pro-Zeichen-Metrik—fahren wir einen Sweep über eine Familie von Mechanismen, die sich darin unterscheiden, wie—oder ob—Parameter angepasst werden: rein lokale Update-Regeln auf einer gemeinsamen assoziativen Architektur (Hebbsche Regel, Oja-Regel, Delta-Regel), eine energiebasierte lokale Regel (Equilibrium Propagation), ein gradiententrainiertes rekurrentes Substrat mit Relaxationsdynamik und pro Einheit gelerntem Vergessen, ein parametergleicher Decoder-only-Transformer sowie zwei parameterfreie, direkt aus den Daten gewachsene Kontextmodelle (Context-Tree Weighting und ein Kontextgraph variabler Ordnung).

Das Ergebnis ist nicht das, was das Substrat-Programm erwartete. Die lokalen assoziativen und energiebasierten Regeln lernen keine Sprache: Hebbsche Regel, Oja-Regel und Equilibrium Propagation sitzen alle an oder über der Buchstabenhäufigkeits-Schranke, und die fehlerkorrigierende Delta-Regel taucht nur knapp darunter. Das gradiententrainierte Substrat ist mit dem Transformer konkurrenzfähig und schlägt ihn auf dieser Skala sogar. Doch das einzelne beste Modell auf dem 107k-Zeichen-Korpus ist der parameterfreie Kontextgraph mit 2.44 Bits/Zeichen, vor dem Gradienten-Substrat (2.70), Context-Tree Weighting (2.73) und dem parametergleichen Transformer (3.36)—ganz ohne gelernte Parameter. Zwei kontrollierte Experimente stimmen in der negativen Hälfte des Bildes überein: Auf einer Aufgabe zur kompositionalen Generalisierung erreicht eine lokale Predictive-Coding-Regel 45%, wo Backpropagation unter identischer Architektur 83% erreicht, und eine Sonde zum kontinuierlichen Lernen zeigt katastrophales Vergessen in jedem Modell. Der korrigierte Schluss ist, dass kleinskalige Sprachfähigkeit hier aus einer expliziten *Kontext*-Struktur variabler Ordnung kommt—erreichbar durch Gradientenabstieg, aber am effizientesten durch einen datengewachsenen Kontextgraphen gewonnen—und nicht aus der festen assoziativen Dynamik des Substrats. Die architektonische Konsequenz ist, die zertifizierte Hebbsche Engine als verteilte Wissens- und Konsensschicht zu behalten, während die Sequenz-Komponente auf einen datengewachsenen Kontextgraphen zeigt—die Richtung, die wir in weiterführender Arbeit aufnehmen.

Geltungsbereich dieses Papers: Die folgenden Experimente sind bewusst klein—zeichenbasierte Modelle mit $\sim 10^5$ Parametern (oder keinen), trainiert auf 10^5–10^6 Zeichen auf einer einzelnen Workstation. Sie sind keine Behauptungen über Verhalten auf Frontier-Skala. Ihr Zweck ist vergleichend und mechanistisch: den Beitrag des *Mechanismus* zu isolieren, indem Architektur, Daten und Evaluation festgehalten werden. Absolute Bits-pro-Zeichen-Werte sind nur relativ zueinander und zu den genannten Baselines zu lesen. Wo eine parameterfreie Methode eine gelernte übertrifft, gilt die Behauptung spezifisch für diese Skala und diesen Korpus; die Bitter Lesson [12] betrifft das entgegengesetzte, asymptotische Regime, und wir adressieren die Spannung explizit in der Diskussion.

1 Einleitung

Das hier untersuchte Substrat entstammt derselben Festlegung wie die Emergente Graphentheorie: dass Funktion von einem *dynamischen* Graphen von Korrelationen zwischen Elementen getragen wird statt von festen paarweisen Gewichten—der Sichtweise, die in der Korrelationstheorie der Hirnfunktion formalisiert ist [1]. In der assoziativen Engine erscheint dies als Bindung durch Phase, Recall durch Relaxation in einen Attraktor und Lernen durch Hebbsche Kovarianz, mit einem Resonanz-Zerfall, der ungenutzte Struktur verblassen lässt. Die Engine ist per Konstruktion dezentral und deterministisch.

Daraus folgt eine natürliche Hoffnung: dass ein solches Substrat, genügend Text ausgesetzt, Sprache aus seiner eigenen Dynamik erwerben würde—ohne das globale, biologisch unplausible Credit Assignment der Backpropagation. Dieses Paper prüft diese Hoffnung direkt und weitet die Prüfung dann aus. Wir trennen zwei Dinge, die leicht vermengt werden. Das eine ist die *Architektur*: Relaxation, Vergessen, Phasenbindung. Das andere ist die *Lernregel*: wie Parameter aus dem Vorhersagefehler angepasst werden. Indem wir dieselbe Architektur unter verschiedenen Lernregeln auf denselben Daten fahren und einen Transformer als externe Referenz aufnehmen, können wir fragen, wo die Fähigkeit lebt. Dann fügen wir eine dritte Achse hinzu, die die ursprüngliche Dichotomie auslässt—die *Modellklasse*—, indem wir parameterfreie Kontextmodelle aufnehmen, die aus den Daten gewachsen sind, statt überhaupt trainiert zu werden. Das erlaubt uns zu fragen, nicht nur ob der Gradient oder die Dynamik die Fähigkeit trägt, sondern ob überhaupt eines von beiden notwendig ist.

2 Experimenteller Aufbau

Gemeinsames Harness. Alle Modelle teilen sich eine Schnittstelle. Text wird auf Zeichenebene tokenisiert, mit einem aus dem Korpus gelesenen Vokabular; kein externer Tokenizer kommt zum Einsatz. Der Korpus wird in einen Trainingsanteil und ein held-out-Validierungsende geteilt. Die Leistung wird als Validierungs-Bits-pro-Zeichen (bpc) berichtet, die Kreuzentropie der Vorhersage des nächsten Zeichens in Bits, gemessen auf Text, auf dem das Modell nie trainiert. Der Split macht die Unterscheidung zwischen Memorisierung und Generalisierung sichtbar: Der Trainingsverlust allein ist uninformativ, da ein hinreichend ausdrucksstarkes Modell ihn auf einem kleinen Korpus gegen null treibt, während der Validierungsverlust steigt. Drei Referenzpunkte verankern jeden Lauf: ein Gleichverteilungscode ($\log_2 V$), ein Unigramm-Code (Buchstabenhäufigkeit) und ein Bigramm-Code. Wir behandeln „unter der Bigramm-Schranke" als Schwelle für genuine Kontextnutzung und „an oder über der Unigramm-Schranke" als Abwesenheit jedes Sprachsignals.

Modell A — gradiententrainiertes Substrat. Ein rekurrenter Kern, in dem sich der verborgene Zustand in jedem Schritt durch eine Leaky-Relaxation in Richtung eines tanh-Attraktors einschwingt und in dem jede Einheit eine gelernte Retention $\lambda \in (0, 1)$ trägt, die als Resonanz-/Vergessens-Zeitkonstante wirkt. Trainiert wird er mit trunkierter Backpropagation Through Time. Dieses Modell verkörpert die *Dynamik* des Substrats (Relaxation, Vergessen) unter einer *Gradienten*-Lernregel.

Modell B — Transformer. Ein Decoder-only-Transformer mit kausaler Multi-Head-Self-Attention, Pre-Norm-Residualblöcken, einem Positions-Embedding und Next-Token-Vorhersage [2]. Seine handgeschriebenen Gradienten wurden gegen finite Differenzen verifiziert (schlechtester relativer Fehler 1.2×10^{-5}), sodass er eine korrekte Baseline ist, kein Strohmann.

Modell C — native assoziative Engine und die Familie der lokalen Regeln. Der treue Mechanismus des zertifizierten Substrats, *ohne Backpropagation*. Zustand und Konzepte sind komplexe Einheits-Phasor-Vektoren; Ähnlichkeit ist der Realteil des inneren Produkts (Phasenbindung, in der Art holographischer/vektorsymbolischer Speicher [3]). Recall ist eine Relaxation—ein einzelner assoziativer Durchlauf $W\mathbf{h}$ auf der beschränkten Energieschale. Das assoziative Gedächtnis W wird online durch Hebbsche Außenprodukt-Akkumulation von (Nächst-Code, Kontext)-Paaren mit einem Zerfallsterm aufgebaut (Resonanz-Vergessen). Kalibriert wird nur eine einzelne Readout-Temperatur; das Gedächtnis selbst wird nie per Gradient trainiert. Das ist „Vergessen und Hebbsche Kovarianz als Optimierer", die eigene Lernregel des Substrats. Um dieselbe Architektur und feste Zufallscodes herum fahren wir außerdem zwei weitere lokale Regeln—*Ojas* normalisierte Hebbsche Regel und die fehlerkorrigierende *Delta*-Regel (Widrow–Hoff)—sowie, separat, *Equilibrium Propagation* [6], eine energiebasierte lokale Lernregel aus derselben biologisch motivierten Nicht-Backprop-Familie. Diese lassen uns fragen, ob das Scheitern des Substrats, falls es scheitert, spezifisch für die reine Hebbsche Akkumulation ist

oder allgemein für lokales Lernen gilt.

Parameterfreie Kontextmodelle. Schließlich nehmen wir zwei Methoden *ohne gelernte Parameter* auf, direkt aus den Daten gewachsen. *Context-Tree Weighting* (CTW) [7] führt Krichevsky–Trofimov-Zählungen über alle Suffix-Kontexte beschränkter Tiefe und mischt ihre Vorhersagen durch Bayessche Gewichtung; wir fahren es sowohl symbolnativ (Tiefe 8) als auch über den rohen Bitstream. *CTXGraph* ist ein Kontextgraph variabler Ordnung (Tiefe ≤ 4): eine Verallgemeinerung des Kontextbaums, in der Kontextzustände geteilt werden können, statt strikt suffix-verschachtelt zu sein, mit KT-artigen Zählungen und einer optionalen *Erhaltungs*-Nebenbedingung an der Gewichtung. Wie CTW hat er keine trainierten Parameter; seine Struktur wächst aus dem Korpus, im Geist der Kontextmodellierung variabler Ordnung [8]. Diese Methoden sitzen auf demselben Harness und werden mit derselben held-out-bpc bewertet.

Die Modelle A und B sind an zwei Betriebspunkten parametergleich (eine kleinere Konfiguration mit $\sim$7–9 $\times 10^4$ für den breiten Sweep und eine größere mit $\sim$2.2–2.4 $\times 10^5$ für den Zwei-Korpus-Vergleich) und werden mit derselben Lernrate trainiert. Die Familie der lokalen Regeln läuft bei $\sim 7 \times 10^4$ Parametern; die Kontextmodelle haben keine. Alle Sprachläufe verwenden Early Stopping am Validierungsminimum, da sich jedes parametrische Modell an kleine Korpora überanpasst.

3 Kompositionale Generalisierung

Vor der Sprache isolieren wir die Behauptung, die lokalem, energiebasiertem Lernen am häufigsten angeheftet wird: dass es *kompositional* generalisiert, wo Standardmethoden es nicht tun. Wir verwenden eine minimale SCAN-„add-jump"-Aufgabe beschränkter Länge [4]: Ein Primitiv wird im Training nur isoliert gesehen und muss dann in Modifikator-Kompositionen (TWICE, THRICE) verwendet werden, in denen es nie trainiert wurde. Der Decoder verwendet einen geteilten Ausgabekopf mit gelernten Slot-Positionen, sodass die held-out-Aktion in jeder Position erreichbar ist—andernfalls wäre die Aufgabe per Konstruktion unlösbar.

Wir vergleichen eine lokale Predictive-Coding-Regel mit Backpropagation unter *identischer* Architektur, identischem Verlust und identischen Daten, mit rohen Token-Eingaben und gelernten Embeddings für beide; nur die Gewichts-Update-Regel unterscheidet sich. Gemittelt darüber, welches Primitiv zurückgehalten wird, und über Seeds fitten beide Regeln das Trainingsset vollständig, doch ihre Generalisierung unterscheidet sich scharf: Backpropagation erreicht 83% Exact-Match auf den held-out-Kompositionen, während die lokale Predictive-Coding-Regel 45% erreicht und über Inferenz-Einstellungen hinweg im Bereich 17–45% bleibt. Volle Relaxation schließt die Lücke nicht. Auf dieser Aufgabe verleiht die lokale Regel keinen kompositionalen Vorteil; die kompositionale Fähigkeit, die auftritt, wird von der Architektur getragen (dem geteilten Decoder) und mit gewöhnlicher Backpropagation gewonnen. Das ist konsistent mit dem Ergebnis, dass Predictive Coding die Backpropagation nur in einem Grenzfall approximiert [5], und damit, dass systematische Generalisierung in dieser Familie ohne Meta-Lernen oder eine eingebaute Grammatik schwer ist.

4 Sprachmodellierung

Wir fahren den vollen Mechanismus-Sweep auf einem 107k-Zeichen-Korpus in einem konstruierten/philosophischen Register mit Vokabular 103 (schwerer, lokal irregulär). Tabelle 1 berichtet held-out-Bits-pro-Zeichen am Validierungsminimum, geordnet von der besten zur schlechtesten Methode, mit den drei Referenz-Baselines unterhalb der Linie. Die parametrischen Modelle in diesem Sweep liegen am kleineren Betriebspunkt von $\sim$7–9$\times 10^4$; die Kontextmodelle tragen keine Parameter. Anschließend berichten wir in Tabelle 2 einen separaten, parametergleichen Zwei-Korpus-Vergleich von A, B und C auf der größeren Skala.

Tabelle 2: Parametergleicher Zwei-Korpus-Vergleich auf der größeren Skala von ~ 2.2–2.4×10^5 (Modelle A, B) und der nativen Engine (C). Striche: nicht gefahren.

Korpus (Zeich., Vok.)	A: Subst. (BP)	B: Transf.	C: Hebb.
Codex (107k, 103)	**2.40**	2.69	—
Prosa (291k, 83)	**2.07**	2.20	5.13
Gleichvert.-Baseline	6.69/6.38		

Fünf Beobachtungen folgen daraus.

Erstens, und am folgenreichsten: Das beste Modell auf diesem Korpus hat keine gelernten Parameter. Der Kontextgraph variabler Ordnung (CTXGraph, Erhaltung aus) erreicht 2.44 bpc, vor dem gradiententrainierten Substrat (2.70, $\sim$0.25 Bits), dem symbol-nativen CTW (2.73) und dem parametergleichen Transformer (3.36, $\sim$0.92 Bits). Eine Methode, die aus den Daten *gewachsen* ist, nicht trainiert,

Tabelle 1: Held-out-Bits/Zeichen am Validierungsminimum auf dem 107k-Korpus (Vokabular 103), niedriger ist besser. „Liest Kontext?" wird gegen die Bigramm-Schranke (3.41) beurteilt: *ja* = darunter (genuine Kontextnutzung), *schwach* = nur unter dem Unigramm, *nein* = an oder über der Unigramm-Schranke. Parametrische Modelle liegen am Betriebspunkt ~ 7–9×10^4; Kontextmodelle haben keine gelernten Parameter und sind aus den Daten gewachsen.

Methode	Val. bpc	Gelernte Parameter	Liest Kontext?
CTXGraph (D= 4, conserve aus)	**2.44**	0 (gewachsen)	**ja (Bestwert)**
Substrat (BPTT, Modell A)	2.70	93.9k	ja
CTW (symbol-nativ, D= 8)	2.73	0 (gewachsen)	ja
CTW (binärer Bitstream, D= 16 Bit)	3.11	0 (gewachsen)	ja
Transformer (Modell B)	3.36	80.1k	ja
CTXGraph (D= 4, conserve an)	3.40	0 (gewachsen)	ja ($\approx$ Bigramm)
Delta-Regel (Widrow–Hoff)	4.56	70.7k	schwach
Hebbsche Regel (Modell-C-Familie)	4.81	70.7k	nein
Oja-Regel	4.90	70.7k	nein
Equilibrium Propagation	5.56	79.3k	nein
Baselines: Gleichverteilung 6.69	Unigramm (Buchstabenhäufigkeit) 4.63		Bigramm 3.41

ist der effizienteste Lerner im Vergleich. Dies ist das Ergebnis, das die Erwartung des Substrat-Programms revidiert: Die Fähigkeit erfordert kein Gradienten-Credit-Assignment, und auf dieser Skala ist sie damit auch nicht am besten bedient.

Zweitens: Lokales Lernen lernt keine Sprache. Die Hebbsche Regel (4.81) und die Oja-Regel (4.90) sitzen an der Unigramm-Schranke, und Equilibrium Propagation—eine energiebasierte lokale Regel aus derselben Familie, zu der das Substrat gehört—ist noch schlechter (5.56, über dem Unigramm). Nur die fehlerkorrigierende Delta-Regel zeigt ein schwaches Signal (4.56, knapp unter dem Unigramm), wie für eine Regel zu erwarten, die ein Schritt hin zum Gradientenabstieg ist, nicht von ihm weg. Das negative Ergebnis gilt hier also allgemein für lokales Lernen und ist kein Artefakt der reinen Hebbschen Akkumulation. Das bestätigt und erweitert den Befund zur nativen Engine (Modell C) in Tabelle 2.

Drittens: Die beiden Negative sind nicht dasselbe Negativ. CTXGraph und CTW sind ebenfalls gradientenfrei, ebenfalls parameterfrei—und sie gewinnen. Das Scheitern der Hebbschen Familie liegt also nicht intrinsisch an „keiner Backpropagation" oder an „keinen gelernten Parametern". Es ist spezifisch für den *Mechanismus*: ein assoziatives Außenprodukt-Gedächtnis fester Kapazität unter Interferenz [9], in dem die Zahl überlappender Assoziationen weit übersteigt, was das Substrat ohne gegenseitige Auslöschung halten kann. Die produktive gradientenfreie Struktur ist eine explizite Kontext-Aufzeichnung variabler Ordnung, keine assoziative Matrix.

Viertens: Die Erhaltungs-Nebenbedingung ist teuer. Schaltet man sie ein, kollabiert CTXGraph von 2.44 auf 3.40 bpc—ungefähr die Bigramm-Schranke—, eine Strafe von ~ 0.96 Bits, der Unterschied zwischen genuiner Kontextnutzung und fast keiner. Welche Invariante die Nebenbedingung der Gewichtung auch aufzwingt, sie gibt den Großteil der Fähigkeit des Modells preis, lokale Kontextstatistik auf diesem Korpus zu fitten. Wir berichten dies als gemessene Eigenschaft des Schätzers und extrapolieren es nicht auf die Erhaltungsprinzipien des zugrunde liegenden physikalischen Rahmenwerks, die einen anderen Gegenstand betreffen; die Beziehung ist eine Frage für die tiefere Arbeit, keine Behauptung dieses Papers.

Fünftens: Das Gradienten-Substrat schlägt den Transformer weiterhin knapp, und der Vorsprung schrumpft mit der Skala. Das Substrat schlägt den Transformer um $+0.66$ bpc bei $\sim$90k Parametern (Tabelle 1), um $+0.29$ bei $\sim$220k (Tabelle 2, Codex) und um $+0.13$ bei $\sim$220k auf dem größeren 291k-Korpus (Prosa). Zwei dieser Punkte variieren die Modellgröße und einer die Datengröße; dies ist also eine konsistente *Richtung* über beide Arten von Skala hinweg und keine einzelne saubere Kurve—die Richtung, die zu erwarten ist, wenn der Langreichweiten-Vorteil des Transformers bei kurzem Kontext und wenig Daten weitgehend ungenutzt bleibt und mit wachsender Skala wächst. Die beiden Tabellen liegen an verschiedenen Betriebspunkten; auf der größeren Skala verbessern sich A und B beide (auf dem Codex Substrat $2.70 \to 2.40$, Transformer $3.36 \to 2.69$), und das Substrat behält seinen Vorsprung. Wir behaupten nichts über den gemessenen Trend hinaus; ob sich die Marge des

Substrats bei einem kleinen positiven Wert stabilisiert oder die Null kreuzt, erfordert einen größeren Korpus, und ob CTXGraphs Vorsprung dieselbe Skalierung überlebt, ist die wichtigere offene Frage.

Beispiel-Fortsetzungen decken sich mit den Zahlen: Das Gradienten-Substrat und der Transformer emittieren orthographisch gültige Wörter und lokale Fragmente des Quellregisters ohne kohärente Bedeutung, während die Hebbsche Engine nahezu zufällige Zeichenketten produziert. Keines der parametrischen Modelle ist flüssig; $\sim$2 Bits/Zeichen sind ein bescheidenes Zeichenmodell, und dass ein parameterfreier Kontextgraph dieses Niveau erreicht, ist eine Aussage über Effizienz, nicht über Flüssigkeit.

5 Kontinuierliches Lernen und Vergessen

Eine zweite Motivation, die sich an das Substrat heftet, ist, dass es—anders als ein eingefrorener Transformer—„keinen Endzustand" erreicht und aus jeder Eingabe lernt. Wir prüfen dies mit einem prequentiellen Protokoll—erst vorhersagen, dann aktualisieren—auf einem Strom, der zwischen zwei Stilen alterniert (A und B, wobei B eine Buchstabenpermutation von A mit demselben Alphabet, aber disjunkter Statistik ist), in der Reihenfolge $A\,B\,A\,B$. Feste held-out-Sonden für jeden Stil werden durchgehend gemessen.

Zwei Tatsachen treten hervor. Erstens ist kontinuierliches Online-Lernen *nicht* distinktiv für das Substrat: Ein auf demselben Strom trainierter Transformer lernt ebenfalls mit jeder Eingabe und erreicht ebenfalls keinen Endzustand. „Eingefrorene Gewichte" in produktiv eingesetzten Systemen sind eine operative Entscheidung, keine architektonische Grenze. Zweitens vergisst jedes Modell katastrophal [10]: Wechselt der Strom zu B, steigt der held-out-Verlust auf A in allen Fällen; das Resonanz-Vergessen des Substrats löst das nicht. Das Substrat zeigt durchaus *kleinere* Ausschläge—sein Verlust auf der A-Sonde bewegt sich in einem schmaleren Band als der des Transformers, der jeden Stil tiefer lernt, ihn aber schärfer verliert—, doch das ist mit schwächerer Plastizität konfundiert: Das Substrat erreicht auf jedem Stil auch einen höheren (schlechteren) besten Verlust, es hat also weniger zu verlieren. Alle Modelle zeigen einen „Savings-Effekt" (schnelleres Wiedererlernen bei Rückkehr). Ob selektives Vergessen einen genuinen Stabilitätsvorteil liefert—geringes Vergessen, *ohne* Plastizität zu opfern—, ist durch diese Läufe nicht etabliert und würde verlangen, erst die Plastizität anzugleichen, bevor man das Vergessen vergleicht [11].

6 Diskussion

Die Experimente stimmen im negativen Ergebnis überein und revidieren das positive. Auf einer kompositionalen Aufgabe bleibt die lokale Regel unter identischer Architektur hinter der Backpropagation zurück; auf Sprache scheitern die native Regel des Substrats und lokales Lernen überhaupt daran, über die Buchstabenhäufigkeit hinaus zu komprimieren; und kontinuierliches Lernen, die plausibelste distinktive Fähigkeit des Substrats, teilt der Transformer, und es lässt das katastrophale Vergessen ungelöst. Entfernt man den Gradienten aus dem Substrat und behält seine Dynamik (Modell A $\rightarrow$ Modell C), kollabiert die Leistung von konkurrenzfähig auf nahezu trivial. So weit entspricht das dem schlimmsten Fall des Substrat-Programms.

Doch der Sweep widerlegt den verführerischen Schluss, die Fähigkeit lebe deshalb *im Gradienten*. Das beste Modell auf dem Korpus, CTXGraph, verwendet keinen Gradienten und keine gelernten Parameter und schlägt sowohl das Gradienten-Substrat als auch den parametergleichen Transformer. Die wirksame Variable ist nicht die Anwesenheit eines Gradienten; es ist die induktive Struktur. Eine explizite Kontext-Aufzeichnung variabler Ordnung—ob als gewachsener Kontextgraph realisiert oder durch Gradientenabstieg auf einem rekurrenten Substrat erreicht—liest den Korpus; ein festes assoziatives Außenprodukt-Gedächtnis tut es nicht, gleichgültig, wie es aktualisiert wird. Gradientenabstieg ist ein Weg zur richtigen Struktur; ein datengewachsener Kontextgraph ist ein anderer, und auf dieser Skala ein deutlich parameter-effizienterer.

Dies schärft die Bitter Lesson [12], statt sie zu illustrieren. Die Standard-Lesart—generische Berechnung unter Suche und Lernen schlägt handentworfene Struktur—beschreibt das asymptotische Großskalen-Regime. Unser Regime ist explizit das entgegengesetzte: $\sim 10^5$ Zeichen und $\sim 10^5$ Parameter oder weniger. Hier schlägt ein gut passender struktureller Prior (ein Kontextgraph) ein generisches Gradientenmodell, genau wie es der eigene Skalen-Vorbehalt der Bitter Lesson vorhersagen würde, und konsistent damit, dass die Substrat-versus-Transformer-Marge mit wachsender Skala schrumpft. Die Lektion für das Programm ist nicht „Struktur verliert", sondern „die Struktur muss zur Aufgabe passen": Die Dynamik des Substrats ist ein struktureller Prior *für assoziativen Recall*, was der falsche Prior für Sequenzvorhersage ist; der Kon-

textgraph ist ein struktureller Prior *für Sequenzvorhersage*, was der richtige ist. Die ursprüngliche Festlegung—dass Funktion in einem dynamischen, aus Daten gewachsenen Graphen von Korrelationen lebt und nicht in festen paarweisen Gewichten [1]—wird, so gelesen, von CTXGraph bestätigt und von der festen Hebbschen Matrix falsifiziert, die genau die „festen paarweisen Gewichte" ist, gegen die sich die Festlegung richtete. Wir formulieren dies als konsistente Lesart, nicht als Herleitung: CTXGraph ist ein klassisches Kontextmodell, nichts, das das Rahmenwerk vorhergesagt hätte, und die Verbindung ist eine, die wir in der tieferen Arbeit prüfen, statt sie hier zu behaupten.

Daraus folgt nicht, dass das zertifizierte assoziative Substrat ohne Wert wäre. Seine etablierten Stärken—Dezentralisierung, deterministischer und zertifizierbarer Zustand, inhaltsadressierbarer assoziativer Recall—sind genau die Eigenschaften einer *Gedächtnis- und Koordinationsschicht*, nicht die eines Sprachmodells. Derselbe Hebbsche Mechanismus, der daran scheitert, Sprache zu komprimieren, passt natürlich zu einem verteilten, verifizierbaren Wissensspeicher, den kompetente Sequenzmodelle abfragen und über den sie Konsens erreichen. Die architektonische Konsequenz ist, Belange zu trennen, die das ursprüngliche Programm verschmolz: Sequenzvorhersage und Schlussfolgern aus Modellen mit der richtigen Kontextstruktur; Wissen aus einer Retrieval-Schicht; Dezentralisierung und Zertifizierung aus der assoziativen Engine, die als gemeinsames Substrat zwischen den Knoten wirkt. Innerhalb der parameterfreien, graph-gewachsenen Familie, die das Programm bevorzugt, benennt das Ergebnis auch die Sequenz-Komponente: ein datengewachsener Kontextgraph, nicht die assoziative Matrix. Das ist die Richtung, die die nächste Arbeitslinie aufnimmt.

7 Fazit

Halten wir Architektur, Daten und Evaluation fest und fahren einen Sweep über den Mechanismus, so finden wir, dass das beste kleinskalige Sprachmodell auf diesem Korpus ein parameterfreier, datengewachsener Kontextgraph ist (2.44 bpc), vor einem gradiententrainierten Substrat (2.70), Context-Tree Weighting (2.73) und einem parametergleichen Transformer (3.36). Die native Hebbsche Regel des Substrats und lokales Lernen überhaupt—Hebbsche Regel, Oja-Regel und Equilibrium Propagation—lernen keine Sprache und plateauieren an oder knapp unter der Buchstabenhäufigkeits-Schranke; nur die fehlerkorrigierende Delta-Regel zeigt ein schwaches Signal. Der korrigierte Schluss ist, dass kleinskalige Sprachfähigkeit hier aus einer expliziten Kontextstruktur variabler Ordnung kommt—erreichbar durch Gradientenabstieg, am effizientesten gewonnen durch einen datengewachsenen Kontextgraphen—und nicht aus der festen assoziativen Dynamik des Substrats, noch aus der Anwesenheit eines Gradienten per se. Zwei weitere kontrollierte Experimente—kompositionale Generalisierung und kontinuierliches Lernen—stimmen in der negativen Hälfte überein. Das Ergebnis ist negativ für die starke These, dass Intelligenz aus der nativen Dynamik des Substrats emergiert, und konstruktiv für eine Architektur, in der die zertifizierte dezentrale Engine als Wissens- und Konsensschicht unter einem datengewachsenen Kontextmodell dient. Die natürlichen nächsten Messungen sind, ob der kleinskalige Vorsprung des Kontextgraphen bei größeren Korpusgrößen und gegen einen Langkontext-Transformer überlebt—und welche Invariante die Erhaltungs-Nebenbedingung zu dem Preis kauft, den sie in Rechnung stellt.

Literatur

[1] von der Malsburg, C., "The Correlation Theory of Brain Function", in *Models of Neural Networks II*, E. Domany, J. L. van Hemmen, K. Schulten (eds.), Springer, New York, pp. 95–119 (1994); orig. MPI für Biophysikalische Chemie, Internal Report 81-2 (1981).

[2] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., Polosukhin, I., "Attention Is All You Need", *Advances in Neural Information Processing Systems* **30** (2017).

[3] Plate, T. A., "Holographic Reduced Representations", *IEEE Transactions on Neural Networks* **6**(3), 623–641 (1995).

[4] Lake, B. M., Baroni, M., "Generalization without Systematicity: On the Compositional Skills of Sequence-to-Sequence Recurrent Networks", *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 2873–2882 (2018).

[5] Whittington, J. C. R., Bogacz, R., "An Approximation of the Error Backpropagation Algorithm in a Predictive Coding Network with Local Hebbian Synaptic Plasticity", *Neural Computation* **29**(5), 1229–1262 (2017).

[6] Scellier, B., Bengio, Y., "Equilibrium Propagation: Bridging the Gap between Energy-Based Models and Backpropagation", *Frontiers in Computational Neuroscience* **11**, 24 (2017).

[7] Willems, F. M. J., Shtarkov, Y. M., Tjalkens, T. J., "The Context-Tree Weighting Method: Basic Properties", *IEEE Transactions on Information Theory* **41**(3), 653–664 (1995).

[8] Cleary, J. G., Witten, I. H., "Data Compression Using Adaptive Coding and Partial String Matching", *IEEE Transactions on Communications* **32**(4), 396–402 (1984).

[9] Hopfield, J. J., "Neural Networks and Physical Systems with Emergent Collective Computational Abilities", *Proc. Natl. Acad. Sci. USA* **79**(8), 2554–2558 (1982).

[10] McCloskey, M., Cohen, N. J., "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem", *Psychology of Learning and Motivation* **24**, 109–165 (1989).

[11] Kirkpatrick, J., Pascanu, R., Rabinowitz, N., et al., "Overcoming Catastrophic Forgetting in Neural Networks", *Proc. Natl. Acad. Sci. USA* **114**(13), 3521–3526 (2017).

[12] Sutton, R. S., "The Bitter Lesson", *incompleteideas.net/IncIdeas/BitterLesson.html* (2019).