N-gram Language Models

Felix Pötzsch (580106) Borisov Timofei Simon Furitsch

4. Juli 2024

Zusammenfassung

Dies ist ein Überblick zum Thema N-Gramme im Modul Aktuelle Themen der Informatik im Sommersemester 2024 an der Hochschule für Technik und Wirtschaft Berlin bei Prof. Dr. Tatiana Ermakova. Wir erklären Grundsätzlich was N-Gramme sind und wie mit dieseer Technik umgegangen wird, sowie einfache Verbesserungstechniken wie Smoothing, Back-Off Word etc.. Des weiteren werden wir ein Beispielprogramm aufführen anhand dessen wir nocheinmal die verwendung von N-Grammen verdeutlichen.

Inhaltsverzeichnis

1 Einführung

1.1 Sprachmodelle

Sprachmodelle sind der Versuch beispielsweise natürliche Sprache als mathematisches Modell darstellen zu können. Ihnen liegt meist ein stochastischer Prozess zugrunde. Bekannte Modelle sind:

- RNN (Rekurrente Neuronale Netze)
  - Abhängigkeiten zwischen Wörtern können erfasst werden und so kontextuell, kohärente Sätze gebildet werden.
  - Verschwindender Gradient. Abhängigkeiten in großer Entfernung können nicht mehr erkannt werden.
- LSTM (Long Short-Term Memory)
  - Art von RNN die das Problem des verschwindenden Gradienten durch Speicherzellen lösen.
  - Zelle ermöglicht selektives merken oder vergessen, wodurch langfristige Abhängigkeiten berücksichtigt werden können.
- GRU:
  - Zielen darauf ab, die Architektur von LSTMs zu vereinfachen und gleichzeitig das Problem des verschwindenden Gradienten anzugehen.
  - Weniger Gating- Mechanismen aber dafür Rechenintensiver.
  - Haben eine vergleichbare Leistung wie LSTM.
- Transformer Modelle:
  - GPT
    - Im Gegensatz zu RNN benutzen Transformer Selbstaufmerksamkeitsmechanismen.
    - Transformatoren verarbeiten Wörter parallel, was zu höherer Effizienz führt.
  - GPT4
    - Ist im Gegensatz zu GPT3 (1,76 Billionen Parameter) kein monolithisches Sprachmodell mehr sondern eher eine Mischung aus 8 x 220 Milliarden Parametermodellen.

- Umfangreicher Textkorpus.
- Besonders gut für kohärente und kontextrelevante Texte.
- Kann für Vielzahl von Anwendungen verwendet werden, z.B.: Codegenerierung.

Im Voraus sollten bei diesen Modellen sowohl Kenntnisse über Sprache im Allgemeinen bestehen, sowie über die jeweilige Kommunikationssituation. Die oben genannten Modelle bestehen zudem aus mehreren Teilmodellen. Meist sind das Vokabular, Häufigkeiten und Grammatik. Die oben genannten Modelle treten zudem in Kombination auf. Sprachmodelle lassen sich in 4 verschiedene Sorten einteilen:

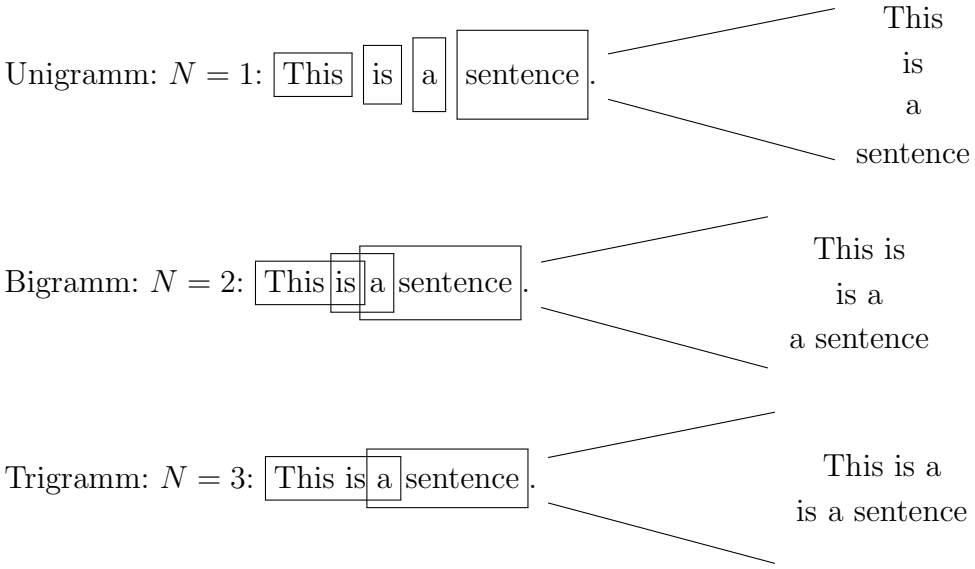
- Regelbasierte Systeme
  - Vordefinierte Sprachregeln, Vorlagen.
  - Fehlende Fähigkeit kontextrelevante dynamische Text zu generieren.
- Vorlagenbasierte Systeme
  - Vordefinierte Vorlagen, die mit dynamischen Inhalten gefüllt werden.
  - Gewisses Maß an Anpassung möglich (Grammatik).
  - Kein einzigartiger kreativer Text.
- Statistische Sprachmodelle
  - N-Gramme, Markov (HMM)
  - Mangel an semantischem Verständnis und Schwierigkeiten kontextrelevante Inhalte zu erzeugen.
- Neuronale Sprachmodelle
  - Dialoge sind möglich und es können sowohl Kontextrelevante als auch Kohärente Teile erkannt werden.
  - Siehe RNN, LSTM, GRU, GPT

1.2 N-Gramme

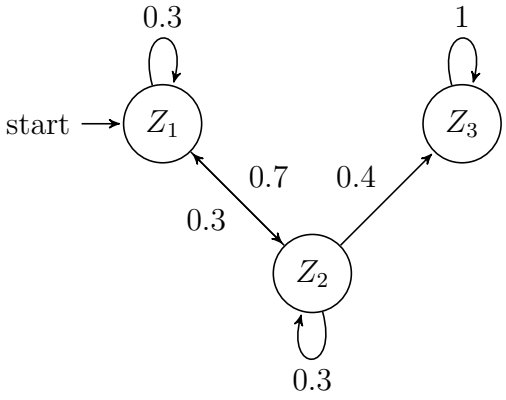
N-Gramme, sind in der Linguistik eine Sequenz von N aufeinanderfolgenden Fragmenten eines Textes. Fragmente können dabei Buchstaben, Silben, Laute oder Wörter sein. Das können von 1-Gramm (Uni-Gramme), über 2-Gramm (Bi-Gramme), 3-Gramm (Tri-Gramme) bis Multigrammen sein. Ihnen zugrunde liegt die Markov-Annahme. Die Markov-Annahme erschließt eine Vorhersage über einen zukünftigen

Zustand nicht aus der Menge aller Zustände, sondern nur der unmittelbaren vorherigen eingetretenen Zustände. Die Zustände können in diesem Falle als Fragmente aufgefasst werden. Die klassischen Anwendungsgebiete für N-Gramme liegen in der Rechtschreibkorrektur, Textvervollständigung/Wörter vorhersage (z.B.: SMS schreiben), Textanalyse (Beurteilung), Spracherkennung (Audioanwendungen) und Handschrifterkennung.

Typische N-Gramme:



Die daraus folgende typische Markov Annahme als Prozessdiagramm und Übergangstabelle würde dazu folgend aussehen:



	Z_1	Z_2	Z_3
Z_1	0.3	0.3	0
Z_2	0.7	0.3	0
Z_3	0	0.4	1
$=$	1	1	1

Grundsätzlich haben wir es mit bedingten Wahrscheinlichkeiten zu tun die wie folgt lauten:

$$\begin{aligned} P(t) = P(w_1, w_2, ...w_n) &= P(w_1) * P(w_2 \mid w_1)...P(w_n \mid w_1, ...w_n - 1) \\ &= \prod_{i=1}^{n} P(w_i \mid w_1...w_i - 1) \end{aligned} \tag{1}$$

$$P(w_1 \mid w_1...w_i - 1) = \frac{P(w_1...w_i)}{P(w_1...w_i - 1)} \tag{2}$$

Also die Wahrscheinlichkeit das Wort 1 eintritt unter der Bedingung, dass ein Paar andere Wörter vorher schon in Kombination aufgetreten sind. Demnach wird Text auf Basis von statistischen Mustern generiert oder anders gesagt: Neu angeordnet. Es mangelt dadurch häufig an semantischem Verständnis. Insbesondere im Zusammenhang mit langfristigeren Abhängigkeiten (Kohärenz) sind Sprachmodelle die rein auf N-Grammen basieren sehr ungeeignet, um Texte zu schreiben. Da lediglich auf Basis einer *Maximum-Likelyhood* Methode gearbeitet wird, kann Text nur *nacherzählt* werden aber nicht generiert.
Es folgen nun ein paar typischen Eigenheiten von N-Grammen. Einem Totalversagen entgegenzuwirken, muss zunächst die Wahl eines geeigneten N priorisiert werden. Denn auch wenn ein geeigneter (Text-) Korpus gut gewählt wurde kann es zu *Unseen N-Grams* kommen. Ebenso wie *Unseen N-Grams* gibt es auch *Unseen/Unknown Words*. Prinzipiell aber sollte gesagt werden, dass auch ein großer oder geeigneter Korpus nicht immer zum Erfolg führt, da auch dort Wortkombinationen oder Wörter *Unseen* sein können (Chomsky, Zipfsches Gesetz). Des Weiteren sollte auf den Underflow bei der Bewertung der Wahrscheinlichkeiten geachtet werden. Dieses Problem führt zum nächsten, denn es müssen nicht nur zu niedrige, sondern auch die vollständigen Nullwahrscheinlichkeiten beachtet werden.
Es müssen also vor allem die Informationen in Form der Angepasstheit sowie Auswahl vorab gut gewählt werden sowie die im Folgenden beschrieben Techniken des *Smoothings*, also der Re-evaluation von Nullwahrscheinlichkeiten und niedrigen Wahrscheinlichkeiten eigesetzt werden. Dazu zählt ebenfalls die zusätzlich im Folgenden beschriebene Perplexität als Evaluationstechnik, um auf *Back Off Words*, also Wörter, auf welche ausgewichen werden kann, zu kommen.

2 Fortgeschrittene Konzepte und Techniken in N-Gramm-Modellen (Borisov Timofei)

2.1 Was ist un-seen N-Grams?

Da jeder Corpus begrenzt ist, fehlen darin zwangsläufig einige völlig akzeptable Wortfolgen. Das heißt, wir werden viele Fälle von vermeintlichen ßero probability n-grams"haben, die in Wirklichkeit eine Nicht-Null-Wahrscheinlichkeit haben sollten [1].

Betrachten wir die Wörter, die auf das Bigram aus Daniel Jurafskys Buch basieren. Ein Textkorpus zusammen mit ihrer Anzahl:

- denied the allegations: 5
- denied the speculation: 2
- denied the rumors: 1
- denied the report: 1

Aber nehmen wir an, unser Testsatz enthält Sätze wie:

- denied the offer
- denied the loan

P(offer—denied the) ist 0! P(loan—denied the) ist 0!

Diese Nullen bedeuten, dass wir die Wahrscheinlichkeit anderer Wortkombinationen stark unterschätzt haben, was die richtige Entscheidung der Anwendung, die auf unserem Modell läuft, stark beeinflussen könnte.

Das Problem heißt "Data Sparsity" [2]. Also kurz gesagt, das bedeutet, dass viele Daten in einem Datensatz fehlen oder auf Null gesetzt sind, sodass die meisten Zellen in einer Tabelle leer bleiben. Dies tritt häufig bei spärlichen Matrizen oder hochdimensionalen Datensätzen auf, wo nicht alle Elemente beobachtet oder erfasst werden. Was können wir also mit den Wörtern machen, die wir verwenden und die nicht in den Kontext unserer Trainingsdaten passen?

Die Methode zur Lösung dieses Problems heißt: Smoothing oder Discounting.

2.2 Smoothing Techniques

2.2.1 Laplace-Glättung. Add One (Add X).

Die einfachste Art der Glättung besteht darin, zu allen n-Gramm-Zählungen einen Wert zu addieren, bevor wir sie zu Wahrscheinlichkeiten normalisieren. Alle Zählungen, die vorher Null waren, haben nun den Wert 1, die Zählungen von 1 werden zu 2 usw. Dieser Algorithmus wird als Laplace-Glättung oder Additive Glätterung bezeichnet [3].

Statt

$$\#(w_1, w_2, w_3, \ldots, w_{n-1}, w_n)$$

Wir setzen

$$\#(w_1, w_2, w_3, \ldots, w_{n-1}, w_n) + 1$$

Da das Vokabular aus V-Wörtern besteht und jedes Wort inkrementiert wurde, müssen wir auch den Nenner anpassen, um die zusätzlichen V-Beschachtungen zu berücksichtigen.

$$P_L(w_n|w_1, w_2, w_3, \ldots, w_{n-1}) = \frac{\#(w_1, w_2, w_3, \ldots, w_{n-1}, w_n) + 1}{\#(w_1, w_2, w_3, \ldots, w_{n-1}) + \mid V \mid}$$

[4]

Eine Alternative zur Glättung durch Addition besteht darin, einen etwas geringeren Anteil der Wahrscheinlichkeitsmasse von den gesehenen zu den ungesehenen Ereignissen zu verschieben. Anstatt zu jeder Zählung 1 zu addieren, fügen wir einen Bruchteil der Zählung k hinzu.

$$P^*_{\text{Add-k}}(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n) + k}{C(w_{n-1}) + kV}$$

[1].

2.2.2 Good-Turing Glättung

Wie berechne ich die Wahrscheinlichkeit von un-seen N-Grams?

Der erste Schritt der Berechnung besteht darin, die Wahrscheinlichkeit abzuschätzen, dass ein zukünftig beobachtetes Fall zu einer bisher nicht gesehenen Art gehört. Diese Schätzung ist:

$$P_{GT}(unseen) = \frac{N_1}{Z}$$

Der nächste Schritt besteht darin, die Wahrscheinlichkeit zu schätzen, dass das nächste beobachtete Fall einer Art angehört, die bereits r-mal beobachtet wurde. Für eine einzelne Art beträgt diese Schätzung:

$$P_{GT}(x) = \frac{r*}{Z}$$

Good-Turing-Smoothing löst dieses Dilemma, indem es die Häufigkeitswerte anpasst. Es wird so getan, als ob Arten, die eigentlich r-mal vorkommen, r^*-mal vorkommen, mit

$$r* = \frac{(r+1)N_{r+1}}{N_r}.$$

[4], [5]

2.2.3 Katz Backoff Glättung

Wenn ein benötigtes n-Gramm in einem Backoff-N-Gramm-Modell keine Vorkommen aufweist, gehen wir zum (n-1)-Gramm über. Dieser Prozess wird fortgesetzt, bis wir eine Sequenz finden, die einige Vorkommen hat.
Damit ein Backoff-Modell eine korrekte Wahrscheinlichkeitsverteilung ergibt, müssen die n-Gramme höherer Ordnung reduziert werden, um etwas Wahrscheinlichkeitsmasse für die n-Gramme niedrigerer Ordnung zu reservieren. Ähnlich wie bei der Add-One-Glättung führt die Verwendung der nicht diskontierten MLE-Wahrscheinlichkeit dazu, dass bei der Ersetzung eines n-Gramms mit einer Wahrscheinlichkeit von Null durch ein n-Gramm niedrigerer Ordnung zusätzliche Wahrscheinlichkeitsmasse hinzugefügt wird. Dies würde dazu führen, dass die Gesamtwahrscheinlichkeit, die das Sprachmodell allen möglichen Zeichenketten zuweist, größer als 1 ist. Neben diesem expliziten Abzinsungsfaktor benötigen wir eine Funktion a, um diese Wahrscheinlichkeitsmasse auf die n-Gramme niedrigerer Ordnung zu verteilen.

Beim Katz-Backoff stützen wir uns auf eine diskontierte Wahrscheinlichkeit P*, wobei wir rekursiv auf die Katz-Wahrscheinlichkeit für das (n-1)-Gramm mit kürzerer Vorgeschichte zurückgehen. Die Wahrscheinlichkeit für ein Backoff n-Gramm PBO wird also wie folgt berechnet:

$$P_{bo}(w_i \mid w_{i-n+1} \cdots w_{i-1})$$

$$= \begin{cases} d_{w_{i-n+1} \cdots w_i} \dfrac{C(w_{i-n+1} \cdots w_{i-1} w_i)}{C(w_{i-n+1} \cdots w_{i-1})} & \text{if } C(w_{i-n+1} \cdots w_i) > k \\ \alpha_{w_{i-n+1} \cdots w_{i-1}} P_{bo}(w_i \mid w_{i-n+2} \cdots w_{i-1}) & \text{otherwise} \end{cases}$$

Zur Berechnung von α ist es sinnvoll, zunächst eine Größe β zu definieren, die die verbleibende Wahrscheinlichkeitsmasse für das $(n-1)$-Gramm darstellt:

$$\beta_{w_{i-n+1} \cdots w_{i-1}} = 1 - \sum_{\{w_i : C(w_{i-n+1} \cdots w_i) > k\}} d_{w_{i-n+1} \cdots w_i} \frac{C(w_{i-n+1} \cdots w_{i-1} w_i)}{C(w_{i-n+1} \cdots w_{i-1})}$$

Anschließend:

$$\alpha_{w_{i-n+1} \cdots w_{i-1}} = \frac{\beta_{w_{i-n+1} \cdots w_{i-1}}}{\sum_{\{w_i : C(w_{i-n+1} \cdots w_i) \leq k\}} P_{bo}(w_i \mid w_{i-n+2} \cdots w_{i-1})}$$

2.3 Vergleich von N-Grammen und neuronalen Netzen.

Im Allgemeinen sind neuronale Netze fortschrittlichere und komplexere Instrumente als N-Gramme. Ihr Vorteil liegt in vielen Aspekten. Zum Bespie: N-Gramme erfassen nur den lokalen Kontext. Ein Trigramm-Modell berücksichtigt nur zwei vorangehende Wörter, um das nächste Wort vorherzusagen. Sie sind nicht in der Lage, weitreichende Abhängigkeiten zu verstehen. Mit steigendem Wert von "n"wächst die Zahl der möglichen n-Gramme exponentiell, was zu hohem Speicherverbrauch und Sparsamkeitsproblemen führt. Probleme mit Wörtern, die mehrere Bedeutungen haben (Polysemie), oder mit verschiedenen Wörtern, die gleich klingen (Homonymie), weil ihnen ein tiefes Verständnis des Kontextes fehlt. N-Grams erfordern manuelles Feature-Engineering, um die relevanten n-Gramme auszuwählen, und führen oft zu hochdimensionalen, spärlichen Vektoren usw.

[6] [7]

Neuronale Netze sind mittlerweile kontextbewusst. Insbesondere mit Architekturen wie LSTM (Long Short-Term Memory) oder Transformer-Modellen können neuronale Netze weitreichende Abhängigkeiten und kontextuelle Informationen über ganze Sätze oder sogar Absätze hinweg erfassen. Sie verfügen über ein automatisches Lernen von Merkmalen während des Trainingsprozesses, wodurch sich der Bedarf an umfangreicher manueller Merkmalstechnik verringert usw.Wir müssen jedoch den Vorteil von

N-Grammen anerkennen. Bei kurzen Texten sind N-Gramm-Methoden dem neuronalen Netz überlegen. [8]

3 Main Section 3 (SIMON)

3.1 Subsection 3.1

Discuss the details of your third main point.

3.2 Subsection 3.2

Further details and analysis related to your third main point.

4 Conclusion

Summarize the main points discussed in your essay and provide your final thoughts.

Literatur

[1] Daniel Jurafsky and James H. Martin, *Speech and Language Processing*, 3rd Edition, Pearson, 2023.

[2] Dremio, "Data Sparsity," `https://www.dremio.com/wiki/data-sparsity/`, Accessed: June 1, 2024.

[3] David Foster, *Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play*, O'Reilly Media, 2020.

[4] Prof. Dr. Johannes Maucher `https://griesshaber.pages.mi.hdm-stuttgart.de/nlp/04ngram/01ngram.html`, Accessed: June 1, 2024.

[5] William A. Gale, "Good–Turing smoothing without tears," *Journal of Quantitative Linguistics*, 1995.

[6] Roshmita Dey , Accessed: June 1, *Understanding Language Modeling: From N-grams to Transformer-based Neural Models* 2024.

[7] Alexander Clark, Gianluca Giorgolo, and Shalom Lappin, *Statistical Representation of Grammaticality Judgements: the Limits of N-Gram Models*, Department of Philosophy, King's College London

[8] A. Suresh Babu, Kumar P.N.V.S.Pavan *Comparing Neural Network Approach with N-Gram Approach for Text Categorization*, January 2010International Journal on Computer Science and Engineering 2(1)