LLMs Unplugged: Teaching Resources for a ChatGPT World

Ben Swift
Australian National University
Canberra, Australia
ben.swift@anu.edu.au

Abstract

Large Language Models (LLMs) are everywhere, yet many learners lack a concrete mental model of how they generate text. This paper presents *LLMs Unplugged*, an unplugged set of activities that teaches the training-to-generation loop (and beyond) using hand-built n-gram models and simple weighted sampling. Workshops based on these resources have been delivered to over 400 participants across secondary, tertiary, and executive-education contexts, and participants report that the activities demystify LLMs by reframing them as probabilistic "next word generation" at scale. All resources are freely available under a Creative Commons license at www.llmsunplugged.org, with a modular design that supports anything from a one hour crash course to a several-day intensive workshop.

CCS Concepts

• **Social and professional topics** → **Computing education**; • **Computing methodologies** → *Natural language processing*; • **Applied computing** → Interactive learning environments.

Keywords

unplugged activities, large language models, n-gram models, AI literacy, hands-on learning

ACM Reference Format:
Ben Swift. 2026. LLMs Unplugged: Teaching Resources for a ChatGPT World. In *28th Australasian Computing Education Conference (ACE 2026), February 09–13, 2026, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3786228.3786237

1 Introduction

For over two decades *CS Unplugged* has shown that core computing concepts can be taught effectively without computers [1]. Through carefully designed hands-on activities, learners from primary school to adult education have learned about algorithms, data structures, and computational thinking. The approach strips away the distractions of syntax and tooling, allowing learners to focus on underlying principles. It works [3], and makes learning this stuff *fun*.

As Machine Learning (in the 2010s) and Artificial Intelligence (in the 2020s) have become more prominent in public discourse, educators have naturally extended the unplugged approach to these CS subfields [11]. There are unplugged activities for teaching classification, clustering, computer vision and artificial neural network (ANN) concepts, showing that even sophisticated AI ideas can be made tangible through physical activities [26].

However, the public release of ChatGPT in November 2022 [18] shifted what "AI" means to most people. Large language models (LLMs) moved from research curiosity to ubiquitous tool almost overnight. Within months, knowledge workers across every domain were using LLMs daily [10], most of whom had no real mental model of how the text they typed into the ChatGPT prompt box produced the text they received as a response.

The existing collections of CS and AI unplugged activities [4] [17, 22] do not contain many resources specifically about language models or text generation. [17] has one "Large Language Mad Libs" activity, but it involves students actually using ChatGPT. [5] has a "cut sentences into words/tokens" activity, but the text generation involves drawing the cut-up words from the bag at random—not a process conducive to high-quality text generation. *CS In Schools* [29] has one activity on Generative AI that uses a "counting bigrams" approach to analysing text, but doesn't show how to generate new text.

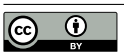
This gap is particularly acute because LLMs have become pervasive far beyond traditional computing education contexts [30]. School-age students certainly need to understand these tools, but so do public servants, executives, journalists, and anyone else who interacts with the world around them via an LLM interface (which increasingly looks like most of us, whether we like it or not). The unplugged approach—building understanding through hands-on activities rather than abstract explanation—seems ideally suited to this broad audience.

This practitioner paper presents *LLMs Unplugged*, a ready-to-use unplugged teaching resource for teaching LLM fundamentals, including:

- a curriculum covering the complete training-to-generation pipeline for language models through hands-on activities
- a modular lesson structure: two Fundamentals lessons (*Training* and *Generation*) and Extensions for deeper exploration
- open materials (CC BY-NC-SA): an online portal at www.llmsunplugged.org with lessons with printable handouts (see Figure 1), instructor notes, and software tools (either web-based or CLI, MIT-licensed) to produce domain-specific n-gram booklets
- practitioner insights from approximately 400 participants across diverse audiences, summarising qualitative reception and delivery patterns

2 What might "LLMs unplugged" look like?

Language modelling has deep historical roots. Andrey Markov's 1913 work on stochastic processes applied to letter sequences in Pushkin's *Eugene Onegin* [12] established the mathematical foundation for modelling language as sequences of dependent random



ACE 2026, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-2352-0/26/02
https://doi.org/10.1145/3786228.3786237

variables. Markov's interest was purely mathematical—proving properties of dependent random variables through empirical text analysis—but his work established that language has statistical structure that can be quantified.

Claude Shannon built directly on this foundation three decades later [24]. Between 1948 and 1951, Shannon applied his new information theory to written English, using n-gram models to measure entropy and redundancy in language. Crucially, Shannon was the first to systematically generate synthetic text using these models, starting with random letters (0-gram), then letter frequencies (1-gram), then letter pairs (2-gram), and progressively higher orders. This generative approach revealed how increasing context length produces increasingly realistic text—a finding that remains central to modern language models.

Markov and Shannon's work was itself "unplugged": counting transitions by hand, calculating probabilities manually, and even generating synthetic text by creating hand-drawn tables and selecting letters based on their frequencies. Modern LLMs use the same fundamental approach—modelling language as weighted distributions over sequences—but at vastly greater scale and with learned rather than hand-crafted statistics. This historical work was influential in the design of *LLMs Unplugged*.

When considering how to teach the fundamental concepts of language models in an unplugged style, several design constraints emerged from *CS Unplugged* design patterns [16] and the specific characteristics of language models.

- End-to-end generation: cover the complete training → generation pipeline. Language models are fundamentally generative; preserving this quality keeps activities engaging and directly relevant to how people use LLM tools.
- Modular and low-friction: each activity should stand alone in a short session, need minimal materials, work for various group sizes, and be easy to adapt. Lessons should build on each other but also be independently useful.
- Broad accessibility: it needs to work for audiences beyond computing students. No programming assumed and minimal mathematics; rely on hands-on activities and plain language.

3 LLMs Unplugged

The *LLMs Unplugged* suite of unplugged activities is available under a Creative Commons license (CC-BY-NC-SA) from www.llmsunplugged.org.

The core mechanic is simple: students build their own n-gram language models (from scratch) using a children's book such as *Dick and Jane* [6] or *Dear Zoo* [2] as the training text. In the training phase students fill out this grid by hand, tracking which words follow which other words through simple tally marks. In the generation phase they use their newly trained "model grid" to iteratively generate text by looking up all possible *next* words (and their relative frequencies) and selecting one at random with a dice roll.

The primary learning outcome is straightforward: students understand that language models—whether ChatGPT or their hand-built version—work the same way. LLMs work by keeping track of the patterns in existing text, then generate new text by repeatedly making random choices weighted by what they've seen before.

The *LLMs Unplugged* website linked above has three main components:

(1) The **lessons** are the core content: each is a self-contained activity which establishes the context (what you'll need, your goal, the key idea), describes the algorithm/procedure, and gives a worked example. Printable handouts are also available for classroom use. See the lesson progression in Section 3.1 for a full list of the concepts covered in each lesson.
(2) The **instructor notes** provide the pedagogical scaffolding. For each lesson, they explain the connection between the activity and modern LLMs, suggest discussion questions to deepen understanding, and provide historical or technical context. These notes help educators without deep AI expertise deliver the material effectively.
(3) (Optional) The resources include an open-source (MIT Licensed) **software tool** to allow educators to create custom n-gram booklets from any text corpus. The tool is written in Rust [13] and uses Typst [15] for typesetting, but there is also a web-based version at www.llmsunplugged.org/tools. For any input text/pdf/docx file the tool will tokenise it, compute n-gram statistics, and create a formatted n-gram booklet. This means educators can pre-train models on domain-relevant text—medical case studies, legal documents, poetry, student essays; whatever best connects with their participants—which can then be used in (almost) all of the lesson activities. Using these software tools is optional; the activities work perfectly well when hand-trained on short texts. But they enable scaling the approach to longer texts and larger vocabularies without requiring students to spend hours tallying word pairs, while still allowing for "unplugged" text generation.

Example lesson handouts for the *Training*, *Generation* and *Pre-trained model generation* are included in an Appendix at the end of this paper; all other handouts are available from the LLMs Unplugged website.

3.1 Lesson progression

The *LLMs Unplugged* lessons are organised into Fundamentals—which should be completed in order—and Extensions that can be selected based on interest and available time.

3.1.1 Fundamentals. These two lessons build on each other and form the core of any *LLMs Unplugged* workshop.

- *Training*: students process text by hand—converting to lowercase, treating punctuation as tokens, then counting word pairs and recording tallies in a grid. This mirrors the core of LLM training: learning is counting patterns in text, and the resulting grid *is* the model.
- *Generation*: students use dice rolls weighted by their grid's counts to sample next words one at a time. The randomness explains why LLMs give different responses to the same prompt—and why the output is often predictable, but sometimes surprising.

These Fundamentals lessons are available in two variants: *grid* and *bucket*. The grid version described above uses paper grids and

dice rolls for weighted sampling, connecting well to probability concepts in the maths curriculum. The bucket version is simpler and more tactile—students cut up the input text (printed on paper) using scissors and organise them into buckets, making it suitable for younger learners or when dice maths would be a distraction. Both variants teach the same core concepts. The www.llmsunplugged.org website has a "toggle" for these lessons to switch between the grid and bucket versions of the activities.

3.1.2 Extensions. These lessons can be done in any order after completing the Fundamentals. Each explores a different aspect of how modern language models work. Most of them require a model that was created as part of the *Training* lesson, and students can use the one they created earlier or swap models amongst themselves.

Scaling up.

- *Pre-trained Generation*: students use a provided booklet (generated via a website widget for any uploaded text) containing a model trained on a larger corpus, generating text without having trained the model themselves. This demonstrates the LLM-as-a-service model: most users never train their own models, they just use ones provided by others.
- *Trigram*: students train a model tracking two-word contexts instead of one (this lesson has both grid and bucket variants). The generated text is "better", but this comes at a cost—significantly more grid rows or buckets to keep track of—illustrating the fundamental context-length tradeoff.

Controlling output.

- *Sampling*: students experiment with temperature (dividing counts before rolling) and truncation strategies (e.g. greedy, no-repeat, alliteration). The same model produces noticeably different outputs, showing that generation control matters as much as training data [7].
- *Beam Search*: multiple students track parallel generation paths on separate papers, pruning to the top candidates after each step. This group activity shows why search strategies matter: beam search finds more coherent sequences than single-path sampling.
- *Tool Use*: students designate people or objects as "tools" with trigger words. When the model generates a trigger, generation pauses while the tool returns a result. This shows that LLMs don't contain all knowledge—they learn *when* to delegate to external sources [21].

Context and meaning.

- *Context Columns*: students add columns for grammatical categories (e.g. after verb, after pronoun, after preposition) and combine these counts with word-specific counts (the standard *Generation* procedure) during generation. This handcrafted attention mechanism previews how transformers learn to weight relevant context [28].
- *Word Embeddings*: students treat each word's row as a vector and calculate distances between rows to build a similarity matrix. Words that behave alike cluster together, revealing that meaning emerges from patterns of usage [14, 20].

Model tuning.

- *LoRA*: students train a small "adaptation grid" (a smaller grid with only a subset of the rows) on new domain text and add its counts to their base model during generation. This shows how one foundation model can spawn thousands of specialised versions through lightweight add-on layers [8].
- *RLHF*: students generate multiple candidate outputs, vote on preferences, then adjust counts (+1 for preferred transitions, −1 for rejected ones). This shifts what "good" means from "matches training data" in the direction of "matches human preferences" [19].
- *Synthetic Data*: students generate synthetic text from their model, train a new model on it, and compare. Patterns degrade across generations as rare words vanish and common phrases dominate, demonstrating why training on AI-generated content risks model collapse [25].

This progression covers key ideas used in frontier LLMs like ChatGPT, Claude or Gemini, albeit at smaller scale. The limited vocabulary and repetitive nature of the children's books makes it feasible to explore all these concepts in an unplugged fashion.

3.2 Example lesson plan

This 90-minute outline covers the two Fundamentals lessons (*Training* and *Generation*) plus the *Pre-trained Generation* extension, and has been successfully delivered to groups from high-school age up and from 5 to 50 participants. The printed handouts associated with these three lesson plans are included in an Appendix, although we encourage educators to visit the website to see all the lesson materials and instructor notes. If we have an additional 30 minutes (2 hours total) then we often add the *Sampling* lesson at the end.

Introduction (15 minutes). Icebreaker: ask participants to consider what it means to "model language", followed by a show-of-hands poll revealing how recently they've used ChatGPT.

Training (20 minutes). The educator demonstrates building a bigram model by reading a few sentences from a children's book and filling in a grid with tally marks showing which words follow which other words. Participants then work in small groups (2–3 people) to train their own models on different pages of text, experiencing firsthand how training means counting patterns. The activity concludes by introducing key terminology (training, model, token, vocabulary) and connecting the hands-on activity to the training of "real" LLMs, giving a sense of the differences in scale between the two processes.

Generation (20 minutes). Using a completed bigram grid the educator demonstrates text generation by looking up possible next words, converting tally counts to dice ranges, and rolling a die to make weighted random selections. Groups then use their newly-trained models to generate sentences, producing (sometimes) surprisingly coherent and/or delightfully nonsensical outputs. Discussion introduces terminology (prompt, completion, prediction) and emphasises that real LLMs generate text through the same iterative sampling process.

Pre-trained Generation (20 minutes). Participants receive printed booklets containing bigram models pre-trained on larger text corpora (typically 5000–10000 words). These booklets allow immediate text generation without manual training. Groups experiment with generating longer passages and compare outputs from models trained on different genres (a fun variation is to not tell participants what the training text was and have them guess). This lesson introduces the concepts of pre-training and foundation models, connecting to how LLMs are trained once on massive corpora then made available for immediate use.

Closing (15 minutes). Summary and reflections: how has this workshop changed how you think about language models? How has this changed how you will use language models?

4 Reception: notes from the field

Over the past year at the Australian National University we have run these *LLMs Unplugged* activities with approximately 400 participants across many sessions in groups of five to fifty. The participants have ranged from school-age to undergraduate students (from all across campus) to senior executives in "executive education" short courses. Interestingly, the majority of participants have been senior leaders in the Australian Public Service—with a range of different expertise and significant interest in understanding AI tools they are being asked to use and evaluate.

This is a practitioner paper rather than a controlled pedagogical study. We have not (yet) conducted pre/post testing of conceptual understanding, run control groups without the intervention, or gathered quantitative learning outcome data. What we can report is qualitative reception across these diverse contexts.

The material is engaging and overwhelmingly well received. Participants consistently report that the hands-on activity helps them to build a new mental model of how LLMs work. The most common insight people articulate is that LLMs are "just" doing probability and randomness at scale—not reasoning, not understanding, but sophisticated pattern matching and weighted sampling. This demystification seems particularly valuable for non-technical participants who may have heard LLMs described in almost magical terms.

The generative aspect matters. People are genuinely delighted when their hand-built model produces a sentence that is both grammatical and surprising. This is not just pedagogically useful—it is emotionally engaging in a way that classification tasks are not. Several learners have reported taking their models home to recreate the activity that evening with their own teenage children. When your bigram model trained on *The Cat in the Hat* [23] generates "fish fish fish red one fish two fish", people laugh and immediately want to generate more text to see what else might emerge.

One consistent observation about delivering this material is that there is an inflection point after the first "shareback" of the newly-generated text. The *Training* lesson is necessary set-up, but the room really starts to buzz during *Generation* when we go around the room and people get to share what came out of their new model. Getting each group to do a "dramatic reading" of their generated text helps here too; the more they ham it up the better. From this point on there are laughs and general good vibes, and the questions they ask about LLMs are often more incisive too. If possible, we recommend facilitators leave enough space for an engaging and playful "shareback" time when conducting the session.

The modular design has proven essential in practice. Most of our deliveries are groups of five to fifty people and cover the Fundamentals lessons plus *Pre-trained Generation*, with *Sampling* added on to the end if we have two hours instead of 90 minutes. We have run all of the Extensions at least once, although some are less "battle-tested" in different classroom settings—something we are planning to rectify in the near future. The ability to scale the content up or down based on available time and audience sophistication has been crucial for the material's adaptability.

5 Next steps

The current material represents the beginning of an *LLMs Unplugged* resource pool, but there is plenty of room for more. As noted above, most delivery focuses on the Fundamentals lessons (*Training* and *Generation*) plus the *Pre-trained Generation* and *Sampling* extensions. These have been successfully run with groups from five to fifty people in a tight 90-minute session (as per example lesson plan above), covering the essential training-to-generation pipeline.

The Extension lessons go deeper, moving progressively closer to how real transformer models work while remaining unplugged. These lessons raise interesting questions about depth in unplugged activities. The CS Unplugged approach excels at providing intuitive introductions to complex concepts—"aha" moments that establish foundational understanding [1]. But there is tension between the constraints of unplugged activities (manual computation, limited scale, short timeframe) and exploring concepts in depth. At some point, does meaningful engagement with the ideas require moving beyond the unplugged format?

Looking forward, we will continue to develop and deliver these lessons, especially through teacher training. The entire project—lessons, printable handouts, instructor notes, and software tools—will remain freely available under a Creative Commons BY-NC-SA 4.0 license at www.llmsunplugged.org. We encourage educators to use, adapt, and improve the material, and we are interested in hearing about implementations in different contexts.

Developing a mental model of how LLMs work should not require a computer science degree or months of study. These unplugged activities demonstrate that the core concepts are accessible to anyone willing to spend an afternoon with pen, paper, and dice or scissors. As LLMs become increasingly central to how we work with text and interact with digital systems, this kind of hands-on understanding becomes not just pedagogically valuable but practically necessary.

LLMs Unplugged is not on its own going to save us from the impending epistemological polycrisis [27] or stop people falling in love with chatbots [9]. Still, better and more widespread understanding of what LLMs are (and aren't) is critical if we want people to make better informed decisions about appropriate use and critical evaluation of their outputs.

Acknowledgments

Many thanks to Eddie Aloise King and Cole Cooney for their input and help facilitating these activities over the past year.

References

[1] Tim Bell and Jan Vahrenhold. 2018. CS Unplugged—How Is It Used, and Does It Work? In *Adventures Between Lower Bounds and Higher Altitudes: Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday*, Hans-Joachim Böckenhauer, Dennis Komm, and Walter Unger (Eds.). Springer International Publishing, Cham, 497–521. doi:10.1007/978-3-319-98355-4_29

[2] Rod Campbell. 1982. *Dear Zoo: A Lift-the-Flap Book.* Blackie, London.

[3] Pei Chen, Daner Yang, Ahmed Hosny Saleh Metwally, Jari Lavonen, and Xudong Wang. 2023. Fostering computational thinking through unplugged activities: A systematic literature review and meta-analysis. *International Journal of STEM Education* 10, 1 (2023), 47. doi:10.1186/s40594-023-00434-7

[4] Computer Science Education Research Group, University of Canterbury. [n. d.]. CS Unplugged. https://www.csunplugged.org/en/ Collection of freely available learning activities that teach computer science through engaging games and puzzles without using computers.

[5] Luke Connelly, Karl-Emil Kjær Bilstrup, and Marianne Graves Petersen. 2025. Beyond LLMs as Black Boxes: Activities and an Educational Tool Supporting Unplugged and Digital AI Learning Activities for K-12 Classrooms. In *Adjunct Proceedings of the Sixth Decennial Aarhus Conference: Computing X Crisis.* ACM, Aarhus N Denmark, 1–4. doi:10.1145/3737609.3747112

[6] William S. Gray and Zerna Sharp. 1946. *Fun with Dick and Jane.* Scott, Foresman and Company, Chicago.

[7] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The Curious Case of Neural Text Degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020.* OpenReview.net. https://openreview.net/forum?id=rygGQyrFvH

[8] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022.* OpenReview.net. https://openreview.net/forum?id=nZeVKeeFYf9

[9] Han Li and Renwen Zhang. 2024. Finding Love in Algorithms: Deciphering the Emotional Contexts of Close Encounters with AI Chatbots. *Journal of Computer-Mediated Communication* 29, 5 (Aug. 2024), zmae015. doi:10.1093/jcmc/zmae015

[10] Zhehui Liao, Maria Antoniak, Inyoung Cheong, Evie Yu-Yen Cheng, Ai-Heng Lee, Kyle Lo, Joseph Chee Chang, and Amy X. Zhang. 2025. LLMs as Research Tools: A Large Scale Survey of Researchers' Usage and Perceptions. In *Second Conference on Language Modeling, COLM 2025, Montreal, Canada, October 7-10, 2025.* OpenReview.net. https://openreview.net/forum?id=p0BwJk3R1p

[11] Annabel Lindner, Stefan Seegerer, and Ralf Romeike. 2019. Unplugged Activities in the Context of AI. In *Informatics in Schools. New Ideas in School Informatics*, Sergei N. Pozdniakov and Valentina Dagienė (Eds.). Springer International Publishing, Cham, 123–135. doi:10.1007/978-3-030-33759-9_10

[12] David Link. 2006. Chains to the West: Markov's Theory of Connected Events and Its Transmission to Western Europe. *Science in Context* 19, 4 (Dec. 2006), 561–589. doi:10.1017/S0269889706001062

[13] Nicholas D. Matsakis and II Klock, Felix S. 2014. The Rust Language. In *Proceedings of the ACM SIGAda Annual Conference on High Integrity Language Technology.* ACM, Portland, OR, USA, 103–104. doi:10.1145/2663171.2663188

[14] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings.* http://arxiv.org/abs/1301.3781

[15] Laurenz Mädje. 2022. *A Programmable Markup Language for Typesetting.* Master's thesis. Technische Universität Berlin, Berlin, Germany. https://laurmaedje.github.io/programmable-markup-language-for-typesetting.pdf

[16] Tomohiro Nishida, Susumu Kanemune, Yukio Idosaka, Mitaro Namiki, Timothy C. Bell, and Yasushi Kuno. 2009. A CS Unplugged Design Pattern. In *Proceedings of the 40th ACM Technical Symposium on Computer Science Education.* ACM, Chattanooga, TN, USA, 231–235. doi:10.1145/1508865.1508951

[17] Northwestern University. [n. d.]. AI Unplugged. https://sites.northwestern.edu/aiunplugged/ Collection of no-computer AI literacy activities for middle-school students, including lesson plans on machine learning, data privacy, and AI ethics. Funded by NSF DRL 2343693.

[18] OpenAI. 2022. ChatGPT. https://openai.com/index/chatgpt/ Conversational AI system launched November 30, 2022.

[19] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training Language Models to Follow Instructions with Human Feedback. In *Advances in Neural Information Processing Systems*, Vol. 35. Curran Associates, Inc., 27730–27744. https://proceedings.neurips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html

[20] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* Association for Computational Linguistics, Doha, Qatar, 1532–1543. doi:10.3115/v1/D14-1162

[21] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. In *Advances in Neural Information Processing Systems*, Vol. 36. Curran Associates, Inc., 68539–68551. https://proceedings.neurips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html

[22] Stefan Seegerer and Annabel Lindner. [n. d.]. AI Unplugged. https://www.aiunplugged.org Activities and teaching material on artificial intelligence, including downloadable board games and activities in multiple languages.

[23] Dr. Seuss. 1957. *The Cat in the Hat.* Random House, New York.

[24] Claude E. Shannon. 1951. Prediction and Entropy of Printed English. *Bell System Technical Journal* 30, 1 (Jan. 1951), 50–64. doi:10.1002/j.1538-7305.1951.tb01366.x

[25] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross Anderson. 2024. AI Models Collapse When Trained on Recursively Generated Data. *Nature* 631, 8022 (July 2024), 755–759. doi:10.1038/s41586-024-07566-y

[26] Yukyeong Song, Xiaoyi Tian, Nandika Regatti, Gloria Ashiya Katuka, Kristy Elizabeth Boyer, and Maya Israel. 2024. Artificial Intelligence Unplugged: Designing Unplugged Activities for a Conversational AI Summer Camp. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1.* ACM, Portland OR USA, 1272–1278. doi:10.1145/3626252.3630783

[27] Anna Strasser. 2025. Why a Careless Use of AI Tools May Contribute to an Epistemological Crisis. *P&D - Philosophy & Digitality* 2, 1 (2025), 162–184. doi:10.18716/pd.v2i1.11662

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, Vol. 30. Curran Associates, Inc., Red Hook, NY, USA, 5998–6008. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[29] Hugh E. Williams, Selina Williams, and Kristy Kendall. 2020. CS in Schools: Developing a Sustainable Coding Programme in Australian Schools. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education.* ACM, Trondheim, Norway, 321–327. doi:10.1145/3341525.3387422

[30] Peng Zhang and Gemma Tur. 2024. A Systematic Review of ChatGPT Use in K-12 Education. *European Journal of Education* 59, 2 (June 2024), e12599. doi:10.1111/ejed.12599

Appendix: lesson cards

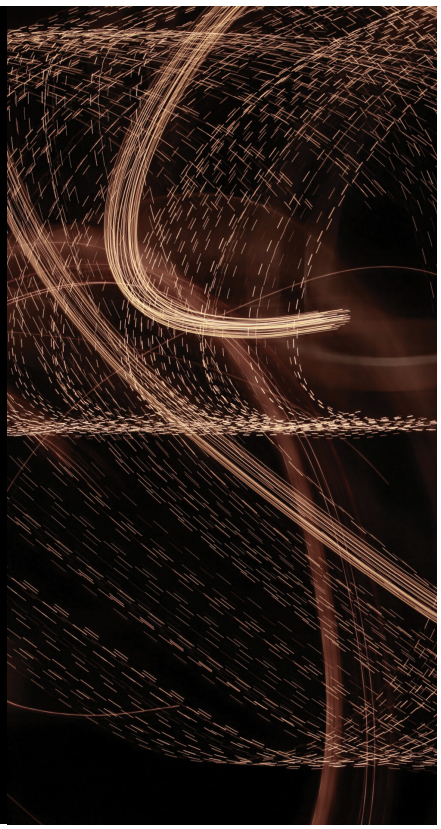
The following pages show printable lesson handouts for the lesson plan described in Section 3.2. Each card is designed to be printed on double-sided A4 paper and includes the lesson goal, required materials, step-by-step procedure, and a worked example. The full set of handouts (as well as instructor notes, interactive demo widgets, and tools for making custom "pre-trained language model" booklets) can be found on the www.llmsunplugged.org website.

Australian National University

LLMs Unplugged | Cybernetic Studio

Training

Build a bigram language model that tracks which words follow which other words in text.



You will need

- some text (e.g. a few pages from a kids book, but can be anything)
- pen, pencil and grid paper

Your goal

To produce a grid that captures the patterns in your input text data. This grid is your *bigram language model*. **Stretch goal**: keep training your model on more input text.

Key idea

Language models learn by counting patterns in text. "Training" means building/constructing a model (i.e. filling out the grid) to track which words follow other words.

LLMs Unplugged | Cybernetic Studio

Algorithm

1. **preprocess your text**:
   - convert everything to lowercase
   - treat words, commas and full stops as separate "words" (and ignore all other punctuation and whitespace)
2. **set up your grid**:
   - take the first word from your text
   - write it in both the first row header and first column header of your grid
3. **fill in the grid** one *word pair* at a time:
   - find the row for the first word (in your training text) and the column for the second word
   - add a tally mark in that cell (if the word isn't in the grid yet, add a new row *and* column for it)
   - shift along by one word (so the second word becomes your "first" word)
   - repeat until you've gone through the entire text

Example

Original text: *"See Spot run. See Spot jump. Run, Spot, run. Jump, Spot, jump."*

Preprocessed text: `see` `spot` `run` `.` `see` `spot` `jump` `.` `run` `,` `spot` `,` `run` `.` `jump` `,` `spot` `,` `jump` `.`

After the first two words (`see` `spot`) the model looks like:

	see	spot				
see		|				
spot						

After the full text the model looks like:

	see	spot	run	.	jump	,
see		||				
spot			|		|	||
run				||		|
.	|		|		|	
jump				||		|
,		||	|		|	

© 2025 Ben Swift | v1.5.0 | CC BY-NC-SA 4.0

Australian National University

LLMs Unplugged | Cybernetic Studio

Generation

Use a pre-trained model to generate new text through weighted random sampling.

You will need

- your completed bigram model (i.e. your filled-out grid) from *Grid Training*
- d10 (or similar) for weighted sampling
- pen & paper for writing down the generated "output text"

Your goal

To generate new text from your bigram language model. **Stretch goal**: keep going, generating as much text as possible. Write a whole book!

Key idea

Language models generate text by predicting one word at a time based on learned patterns. Your trained model provides the "next word" options and their relative probabilities; dice rolls provide the randomness to choose one of those options (and this process can be repeated indefinitely).

LLMs Unplugged | Cybernetic Studio

Algorithm

1. **choose a starting word** — pick any word from the first column of your grid
2. **look at that word's row** to identify all possible next words and their counts
3. **roll dice weighted by the counts** (see the *Weighted Randomness* lesson)
4. **write down the chosen word** and use that as your next starting word
5. **repeat** from step 2 until you reach the desired length *or* a natural stopping point (e.g. a full stop `.`)

Example

Using the same bigram model from the example in *Grid Training*:

	`see`	`spot`	`run`	`.`	`jump`	`,`
`see`		||				
`spot`			|		|	||
`run`				||		|
`.`	|		|		|	
`jump`				||		|
`,`		||	|		|	

- choose (for example) `see` as your starting word
- `see` (row) → `spot` (column); it's the only option, so write down `spot` as next word
- `spot` → `run` (25%), `jump` (25%) or `,` (50%); roll dice to choose
- let's say dice picks `run`; write it down
- `run` → `.` (67%) or `,` (33%); roll dice to choose
- let's say dice picks `.`; write it down
- `.` → `see` (33%), `run` (33%) or `jump` (33%); roll dice to choose
- let's say dice picks `see`; write it down
- `see` → `spot`; it's the only option, so write down `spot`... and so on

After the above steps, the generated text is *"see spot run. see spot"*

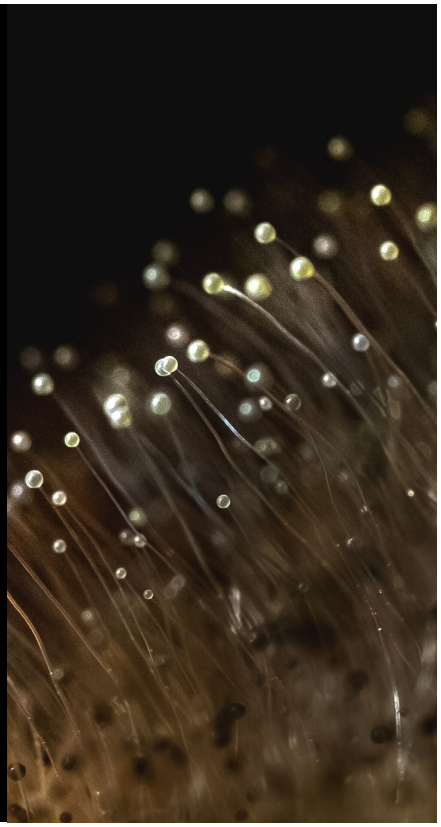
© 2025 Ben Swift | v1.5.0 | CC BY-NC-SA 4.0

Australian National University

LLMs Unplugged | Cybernetic Studio

Pre-trained Model Generation

Use a (slightly larger) pre-trained model to generate new text through weighted random sampling.



You will need

- a pre-trained model booklet
- d10 for weighted sampling
- pen & paper for writing down the generated "output text"

Your goal

To generate new text using a pre-trained language model without having to train it yourself. **Stretch goal**: without looking at the title, try and guess which text the booklet model was trained on.

Key idea

You don't need to train your own model to use one. Pre-trained models capture patterns from large amounts of text and can be used to generate new text just like your "hand-trained" model from *Grid Training*.

LLMs Unplugged | Cybernetic Studio

Algorithm

Full instructions are at the front of the pre-trained model booklet, but here's a quick summary:

1. **choose a starting word** — pick any bold word from the booklet and write it down
2. **look up the word's entry** (i.e. use the booklet like a dictionary) to find all possible *next* words according to the model
3. **roll your d10s** (if required): check for diamonds next to the word — this shows how many d10s to roll (e.g. ♦♦♦ means roll 3 d10s). If there are no diamonds, there's only one possible next word — skip to step 5. Read the dice from left to right as a single number (e.g. rolling 2, 1 and 7 means your roll is 217)
4. **find your next word**: scan through the followers until you find the first number ≥ your roll, or just use the single word if no dice were rolled (write it down)
5. repeat from step 2 using this word as your new word, continuing this loop until you reach a natural stopping point (like a period) or reach your desired text length

Example 1: single d10

Your current word is **"cat"** and its entry shows:

cat ♦ 4|sat 7|ran **10**|slept

- one diamond (♦) means roll 1 d10
- roll your dice: roll a 6
- find the next word: first number ≥ 6 is 7|ran, so next word is "ran"
- write it down, look it up and continue the process

Example 2: multiple d10s

Your current word is **"the"** and its entry shows:

the ♦♦ **33**|cat **66**|dog **99**|end

- two diamonds (♦♦) means roll 2 d10s
- roll your dice: roll 5 and 8 → combine them to get 58
- find the next word: first number ≥ 58 is **66**|dog, so next word is "dog"
- write it down, look it up and continue the process

LLMs Unplugged | Cybernetic Studio

© 2025 Ben Swift | v1.5.0 | CC BY-NC-SA 4.0