Approaches to automatic detection of machine-generated text

Aron Winkler
University of Tuebingen
MAT 6189673
aron.winkler@student.uni-tuebingen.de

Abstract

Recent developments in Natural Language Processing (NLP) have resulted in the development and popularization of highly effective Large Language Models (LLMs), capable of generating convincing and creative linguistic material. LLMs have garnered much attention, both from researchers and the general public, and continue to be increasingly applied to a variety of fields. However, the breakneck speed at which these systems are adopted leaves unattended some of the security concerns regarding their use. Since the language produced by the models is of such high quality, it isn't always feasible to distinguish authentically human contributions from machine-generated text, which can enable a multitude of nefarious applications of LLMs. This work explores the history and inner workings of LLMs, how they can be misused, and possible antidotes to the problem of machine-generated text detection, with a careful eye toward a good balance of computational cost and performance of detection strategies.

1 Introduction

In 1951, Gnome Press published Isaac Asimov's *Foundation* (Asimov, 1951), the first title of a trilogy that would go on to become one of the cornerstones of modern science fiction. In the novel, set in the distant future, scientist Hari Seldon predicts the fall of the Galactic Empire, an event that would pave the way to an era of barbarism in the story's fantastical universe.

To preserve humanity's knowledge and technical skills, Hari Seldon establishes the Foundation on an uninhabited planet on the periphery of the Empire, a sort of outpost dedicated to being the home to the archival effort. The novel follows the political and technological adventures of the Foundation and its leaders, with one of the first plot points being the first conflict between the Foundation and a major local power in the periphery, Anacreon, which declared its independence as the Empire's influence in the periphery weakened. Seeking protection from the Empire against Anacreon's expansionary stance, the Foundation hosts a diplomatic emissary from the Empire, a Lord Dorwin, finally obtaining a convoluted treaty between the Empire and Anacreon over their respective spheres of influence.

> "Before you now you see a copy of the treaty between the Empire and Anacreon – a treaty, incidentally, which is signed on the Emperor's behalf by the same Lord Dorwin who was here last week – and with it a symbolic analysis."
>
> The treaty ran through five pages of fine print and the analysis was scrawled out in just under half a page.
>
> "As you see, gentlemen, something like ninety percent of the treaty boiled right out of the analysis as being meaningless, and what we end up with can be described in the following interesting manner:
>
> "Obligations of Anacreon to the Empire: None!"
>
> "Powers of the Empire over Anacreon: None!"
>
> *Isaac Asimov, Foundation,*
> *Part II: The Encyclopedists*

At this point the Foundation's scientists, through a technique they call *"symbolic analysis"*, condense several pages of treaty into a few lines, revealing the hidden meaning behind the layers of legal dissimulation. By doing so, they expose the inability of the dying empire to exert its influence over its own periphery, and they realize that moving forward, they can only rely on themselves, marking perhaps the true starting point of the story in Asimov's *Foundation*.

Despite first reading this passage when I was a teenager, perhaps over a decade ago, these fictional twists stayed with me through the years. They were, after all, my first indirect exposure to the field of computational linguistics and natural language processing (NLP). I remember being mesmerized by the potential of machine computation applied to natural language, in what I would later learn to better define as a mixture of information retrieval and automatic text summarization.

While Asimov's pen definitely hit the mark in predicting some of the most intriguing and successful applications of NLP in the years ahead, what granted computational linguistics perhaps its brightest moment in the limelight was one of its other, albeit related, subfields: language modelling and generation.

1.1 Language modelling

Teaching a machine to understand and produce natural language is intuitively a difficult task. Even if one could reliably collect all ingredients that make up human language, creating a system that emulates it even just well-enough is a very tall order, since there would likely be millions if not billions of cases to consider. Linguists have documented hundreds of languages, each with their own grammar, peculiarities, exceptions, all of which have yet to be described under one common ruleset. Manually building a program from the ground up for even just one language is beyond what current technology is capable of.

The very first chatbot, ELIZA (Weizenbaum, 1966), simulated conversation through pattern matching and substitution, essentially repeating and paraphrasing their interlocutor's statements. While it successfully bypasses the necessity of programming a machine with *intelligence*, such an approach does not result in a system that can be described as creative in any sense. In other words, ELIZA will never write a poem, or surprise their conversation partner with a witty turn of phrase. It would never be able to tell whether May has 30 or 31 days because it has no notion of what *May* and *days* are. Teaching language is, after all, not only an issue of grammar, but one of world knowledge as well.

If *teaching* language to machines as one would to humans is not possible, and rule-based approaches such as ELIZA inevitably reach a bottleneck, then it becomes necessary to adopt a new strategy, rooted in statistics. This new approach consists in the realization that the sentence "he's wearing a circumference jacket" is much less likely to be uttered than "he's wearing a yellow jacket". Extrapolating the pattern, the set of words that can fill the gap in "he's wearing a _____ jacket" is varied, but "yellow" will have a much higher *probability* of showing up than "circumference". Language models are the tools that are employed to estimate these probabilities.

Due to recent innovations in NLP, the phrase "language model" evokes big and expensive systems, trained on huge amounts of data and costing enormous amounts of money to develop. While this is certainly understandable, the label in itself has no presupposition of size or cost. In essence, language models, and NLG (Natural Language Generation) systems in general, break down the massively complex problem of "teaching language to machines", into the more manageable task of "statistically learning what words are likely to follow others". In other words, language models produce next-word (or, more generally, next-token) probabilities based on an input sequence (Gao and Lin, 2004). After this is achieved, the resulting model can be prompted over and over, one word at a time, in order to generate long pieces of text, by a process called autoregressive generation (Lin et al., 2021).

For example, for the completion "fifteen minutes of _____", one would expect a (good) language model to offer words such as "fame" or "overtime". One idea to achieve this is to collect some linguistic data and observe what words follow "fifteen minutes of" and extrapolate a probability distribution from the observed frequencies. So-called *n-gram* language models (Chen and Goodman, 1999) are built in this fashion, with the *n* in *n-gram* specifying the amount of left context taken into consideration.

The simplest of these models, the bigram (2-gram) language model, records co-occurring word pairs in the sample dataset. Consequently, for this model, only the last word of a sequence determines the prediction over the following word. This results in a model that can reliably generate short collocations, such as "Marie *Curie*", but cannot generate coherent sentences, and would likely even fail to offer "fame" as a completion to "fifteen minutes of _____", since the only available context for the prediction is the word "of". To correctly predict "fame", one would need at least a 4-gram language model, which would finally allow for such a "long" context requirement. However, while taking more context tokens into consideration increases the performance of n-gram language models, it does so at a steep (especially memory) cost: for 4-gram LMs with a vocabulary size (i.e., how many words the model knows) of 1000, for example, implementations without optimizations would require the frequency counts for 10^4 n-grams to be accessible for predictions.

Another issue that presents itself with frequency-based models is the handling of unseen n-grams. For example, the 3-gram "duck goose pony" may never come up in the model training data, in which case some near-0 probability is assigned to the sequence (due to smoothing, see for example Chen and Goodman, 1999). While some n-grams will fail to appear due to being ungrammatical or nonsensical like in "duck goose pony", others may be perfectly well-formed but either rare or just absent from the training data due to chance: if "trees need hydration" was never observed, then the model would fail to recognize this n-gram to be more likely than, for example, "trees need circumference" (assuming the latter was also not observed, which is quite likely in organic text). Even the n-gram "roundly faucet knowledge" would be equally is likely as "trees need hydration" if both were never observed in training. While the latter example can be solved by including lower-order n-grams in the probability calculation ("trees need" may have been observed even if "trees need hydration" wasn't, but "roundly faucet" is unlikely to have been observed; see Katz, 1987), the former case requires more subtle knowledge. In order to correctly assess "trees need hydration" as a quite likely n-gram, the model would need to identify the similarity between the words "water" and "hydration", and infer that "trees need hydration" should have higher probability than, say, "trees need virtual", due to "water" and "hydration" being similar.

Due to such limitations, n-gram language models aren't the piece of technology that propelled language modelling to the heights that we associate with it today. The missing piece of the puzzle are neural networks (Anderson, 1995), which when applied to language generation give rise to *neural language models*.

Following the example above, both n-gram and neural language models solve the fundamental of problem of estimating the probability that the word "fate" follows "fifteen minutes of". However, in order to do so, n-gram language models draw upon explicit frequency observations when generating its output, an approach that often fails to consider an adequate amount of context, or to take into account the fact that similar words appear in similar contexts. In contrast, neural language models draw upon their internal parameters - the weights and biases (Anderson, 1995) associated to the various layers of the neural network that makes up the model.

1.2 Neural language modelling

Neural networks are an extremely powerful for many applications across several disciplines. Providing an effective summary of all neural networks is a difficult task, since they manifest themselves in different variations for different tasks. Still, they are generally understood as interconnected layers of *neurons* that map an input vector of numbers to an output vector. Each neuron in the network is made up of a weight, a bias, and an activation function, which are used to transform an input vector to an output number. For the purposes of this work, it is more important to understand the overall network rather than its individual parts: neural models are a way to apply complex transformations to vectors. The *parameters* of the model, i.e. the combined weights and biases of the individual neurons, can be used to encode knowledge in a way that simpler statistical models struggle to achieve.

Above, the example of "trees need water" and "trees need hydration" was briefly discussed. While n-gram models have no structural way to note the similarity between "water" and "hydration", and thus fail to recognize that the admissible contexts for the two words have some overlap, neural networks have been employed to solve this problem exactly because of their capacity to progressively store knowledge. Word embeddings (Selva Birunda and Kanniga Devi, 2021) are a way of representing words as numerical vectors, such words with similar semantics will be close to each other in terms of vector distance. The embeddings for "water" and "hydration" would therefore be closer in vector space than "water" and "dog". Several ways have been developed to derive embeddings from text data — for exmaple, the CBOW (Mikolov et al., 2013) algorithm uses neural networks to predict the missing word given the surrounding context through the use of a neural network. For many iterations, the model is presented with a piece of text with a gap, and the objective of guessing the missing item. With each example, the model parameters are updated, or *nudged* in the correct direction (for information on gradient descent, see Zhang, 2019), i.e. towards a state that are more conducive to the correct prediction. Importantly, among the parameters of the neural network are the embeddings for every word, which are used by the model to compute the final prediction across subsequent layers — these are random at the beginning, but progressively more and more refined. At the end of training, most model parameters are discarded, but the embeddings are kept, and hopefully the result will have captured semantic similarities between the entries.

While the CBOW algorithm discards all model parameters aside from the embeddings, it is naturally possible to train the neural model with the objective of keeping all of them, still taking advantage of the architecture's capacity to learn and store information. This is the basis for modern, highly sophisticated language models, with billions or even trillions of parameters.

1.3 Large Language Models

Language models have a long and intricate history, having been iterated upon from different perspectives and with different architectural approaches. The introduction of the transformer (Vaswani et al., 2023), a model type that for more efficient training over extremely large text data, propelled language model quality forward considerably, to the point that all language models commonly known today follow this architecture. The ability to train models with relatively little expenses allowed models to grow further and further in perplexity, eventually resulting in what we identify today as Large Language Models (LLMs). In terms of research attention, BERT (Devlin et al., 2019) and GPT-2 (Radford et al., 2019) have perhaps been the most resonant examples earlier on.

Bidirectional Encoder Representations from Transformers (BERT) is slightly different from the language models discussed so far, in that its primary purpose is not generation. The word embeddings described above are a powerful tool for obtaining meaningful representations, but have the significant flaw of not being context-aware. For example, it's clear that the word "honey" should have different embeddings between "bees make honey" and "honey, wake up!".

BERT is a solution to this problem: instead of training a model to then only keep the embedding layer (in other words, computing all embeddings *offline*), BERT is a full-fledged language model that evaluates pieces of text as a whole to return their representations *online*. As such, when evaluating "bees make honey" and "honey, wake up!" with BERT, "honey" will have vastly different embeddings to account for the different context.

BERT was developed, in other words, to compute context-aware word representations, hence the name. It cannot be used for language generation, mostly because it's nature as a *bidirectional* model. For the example language model architectures described above, there was an underlying assumption that only the *left* context is visible to the model. This makes intuitive sense: when generating language, only what came before influences the probability distribution of the next word. This is not the case for BERT. Since the objective here is to evaluate finished productions, BERT may use all previous and subsequent tokens at each timestep.

Despite being a fairly recent introduction to the scene, being made available in just 2018, BERT has made big waves in nearly all fields where processing natural language is even tangentially relevant. It even resulted in the birth of the somewhat jokingly named discipline of BERTology, which is meant to convey the detailed analysis of how BERT and its different variations exactly arrive at the high-quality embeddings that they are known for.

In the same year as BERT was published, another historical language model also made its debut: OpenAI's GPT-2. (Radford et al., 2019) GPT-2 lines up closer to the popular idea of what constitutes a language model. It is large in size with 1.5 billion parameters, and was primarily conceived for language generation. In the original paper, the authors highlighted the ability of the model to approach several different problems without explicit training, such as question answering, test summarization, machine translation, and so forth. In this sense, GPT-2 is one of the first successful examples of language models displaying generalized problem-solving skills, that require both articualation and minute world knowledge.

GPT-2 has become more of a baseline than a challenger in the years following its introduction, such was its impact in research applications. It has garnered much attention and analysis, similarly to BERT, a process that also highlighted some of its flaws (see, for example, the GPT-2 unicorn completion). While talk about GPT-2 could not at all be considered undertone, it was perhaps cut short, in that nowadays it is more rarely employed compared to BERT. This is in large part due to its new iteration, GPT-3 (Brown et al., 2020).

1.4 Fears and reactions to LLMs

In 2020, GPT-3 was announced by the company OpenAI, and was made available to the general public through an interface called ChatGPT. With its introduction, the gates were open to the generation of high-quality text through AI. This was perhaps the first example of language model garnering tremendous general attention, not only in academic circles but from the public at large. ChatGPT even experienced service outages due to its servers not being able to handle the astounding traffic they were receiving. The introduction of GPT-3 provided an unprecedented boost to langauge models as a consumer technology, leading to a sort of arms race both in integrating AI into customer-facing products, as well as in model developent itself. In the first wave of AI competition, Facebook's Llama (Touvron et al., 2023) and the open source model Mistral (Jiang et al., 2023) also entered the scene, alongside Google's now discontinued Bard (see for example Fowler, 2023). In a subsequent wave, further developments have reached even higher generation quality, with models such as GPT-4 (OpenAI et al., 2024) and Gemini (Team et al., 2024).

Such developments marked the beginning of language models being commonplace technology. The arts, education, technology, sales, and dozens of other fields have seen major changes to the way they operate, either in alongside or in opposition to artificial intelligence. Education, in particular, was one of the first areas to experience a problematic application of language generation, with a prevailing initial fear that students would opt to have a language model solve their homework for them. Surprisingly, programming also saw the rapid birth of assistive technology, such as Github Copilot (Chen et al., 2021), leading to fears that many software professionals would be made obsolete by the new technologies.

People employed in artistic fields, particularly writers, also took a deeply cautious stance from the beginning with respect to AI. In a landmark development, the 5-month-long strike of the Writer's Guild of America [1] resulted in an agreement which included safeguards for writers against the use of artificial intelligence. [2] Such a stance turned out to be far from unfounded, with how AI-generated content has flooded several internet platforms. As recently as March 2024, even the Google search engine has had to address the issue of unoriginal content, which is in large part meant to target results which contain generated text. [3] At the same time, the search engine itself adopted a new feature called AI Overviews, an integrated AI-generated response to the user's query. [4]

The use of large language models thus has been observed to spark mixed reactions, both emotionally and legislatively. However, what was considered so far in this examination is the *lawful* and (perhaps arguably) moral use of language technology. This does not mean that such technologies cannot be employed with harmful intent — quite the contrary. the discussion about malicious use of language generation has been left on the sidelines in the early years of LM adoption, but worries are progressively making their way to the forefront of the debate, and countermesures are becoming increasingly pursued research objectives.

2 Threats posed by NLG systems

NLG systems, and language models in particular, have emerged as an extraordinarily useful tool in a number of creative and technical fields, but it stands to reason that they would lend themselves to nefarious applications just as well as ethical ones. Following Crothers et al., 2023, several threat models (Shostack, 2014) can be outlined for language modelling.

3 Previous approaches

4 Task 8 at SemEval 2024

5 Discussion of SemEval results

6 Conclusion

7 Acknowledgements

References

James A Anderson. 1995. *An introduction to neural networks*. MIT press.

[1] An article summarising the strike can be viewed at https://www.vox.com/culture/2023/9/24/23888673/wga-strike-end-sag-aftra-contract

[2] See for example https://www.nbclosangeles.com/news/local/hollywood-writers-safeguards-against-ai-wga-agreement/3233064/ for an account of the agreement

[3] Google published a blog post explaining the changes at https://blog.google/products/search/google-search-update-march-2024/

[4] Google published a blog post explaining its new AI integration at https://blog.google/products/search/generative-ai-google-search-may-2024/

I. Asimov. 1951. *Foundation*. Foundation series. Gnome Press.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Mark Chen et al. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Stanley F Chen and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–394.

Evan Crothers, Nathalie Japkowicz, and Herna Viktor. 2023. Machine generated text: A comprehensive survey of threat models and detection methods. *Preprint*, arXiv:2210.07321.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

G. A. Fowler. 2023. Say what, Bard? What Google's new AI gets right, wrong and weird.

Jianfeng Gao and Chin-Yew Lin. 2004. Introduction to the special issue on statistical language modeling.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Slava Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE transactions on acoustics, speech, and signal processing*, 35(3):400–401.

Chu-Cheng Lin, Aaron Jaech, Xin Li, Matthew R. Gormley, and Jason Eisner. 2021. Limitations of autoregressive models and their alternatives. *Preprint*, arXiv:2010.11939.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Preprint*, arXiv:1301.3781.

OpenAI et al. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

S Selva Birunda and R Kanniga Devi. 2021. A review on word embedding techniques for text classification. *Innovative Data Communication Technologies and Application: Proceedings of ICIDCA 2020*, pages 267–281.

Adam Shostack. 2014. *Threat Modeling: Designing for Security*, 1st edition. Wiley Publishing.

Gemini Team et al. 2024. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

Joseph Weizenbaum. 1966. ELIZA — a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45.

Jiawei Zhang. 2019. Gradient descent based optimization algorithms for deep learning models training. *Preprint*, arXiv:1903.03614.