The Evolution of Embeddings

Looking back to the pre-Transformer times.

AVI CHAWLA
AUG 05, 2024

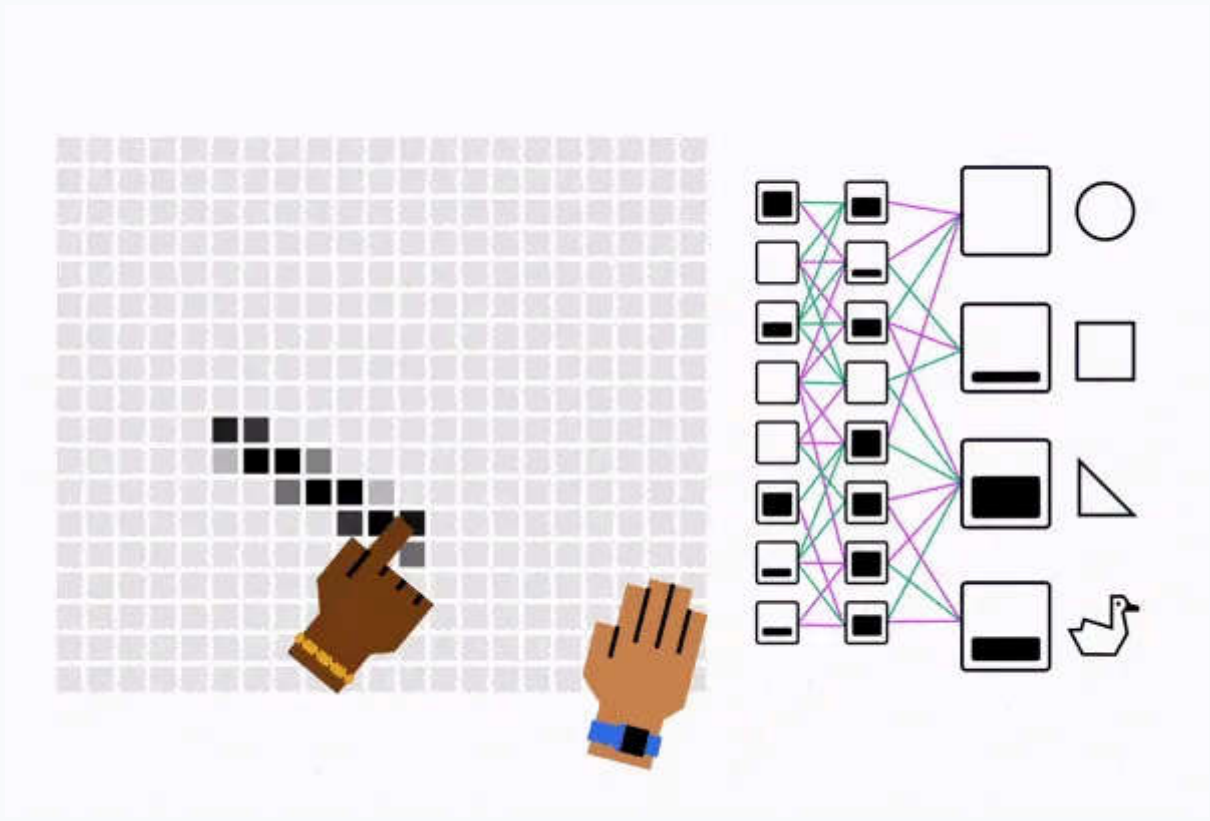
29 4 Share •••

Brilliant — Daily Learning, Lifelong Impact!

This daily newsletter is a result of three key habits I developed during the pandemic, which changed my life forever:

- Learn every day.
- Teach everything I know.
- Work/build in public.

But after leaving university, many still undervalue the importance of consistent learning. Thankfully, you don't need hours of daily commitment to fulfill this goal.



Brilliant is there to help. It is an interactive platform with 1000s of lessons that simplify complex concepts in math, programming, data analysis, etc., using fun, visual experiences. Each lesson is packed with hands-on problem-solving that lets you play with concepts.

With Brilliant, even ten minutes of learning day can help you build a long-term learning habit.

Join over 10 million people around the world by starting your 30-day free trial. Plus, Daily Dose of Data Science readers get a special 20% off a premium annual subscription:

Thanks to Brilliant for sponsoring today's issue.

The Evolution of Embeddings

I often receive emails from readers of this newsletter showing concern about being new to AI and intimidated by stuff like GPTs, Transformers, etc.

In my experience, the problem is quite common because so many people are new to this field and have very little clue about what's happening today.

But the first step in addressing this concern is understanding how we got where we are today — **the foundational stuff.**

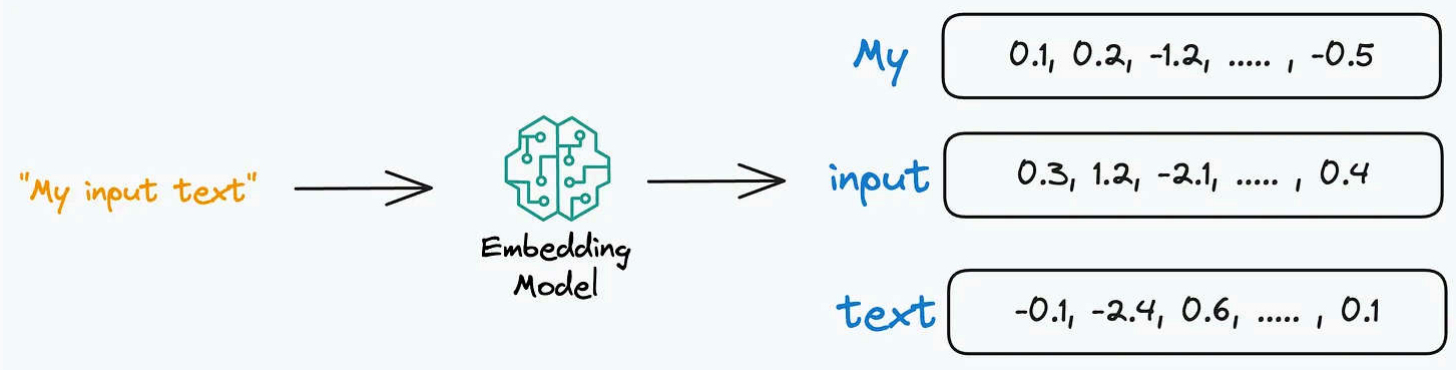
So in today's newsletter issue, I want to step aside from the usual data science-oriented content.

Instead, I want to help you look a decade back into NLP research, understand the pain points that existed back then, how they were addressed, and share some learnings from my research in this domain.

> *Today, we are discussing* ***embeddings*** *but in any upcoming issue, we shall discusse the architectural revolutions in NLP.*

Let's begin!

To build models for language-oriented tasks, it is crucial to generate numerical representations (or vectors) for words.



Text to embedding overview

This allows words to be processed and manipulated mathematically and perform various computational operations on words.

The objective of embeddings is to capture semantic and syntactic relationships between words. This helps machines understand and reason about language more effectively.

In the pre-Transformers era, this was primarily done using pre-trained static embeddings.

Essentially, someone would train embeddings on, say, 100k, or 200k common words using deep learning techniques and open-source them.

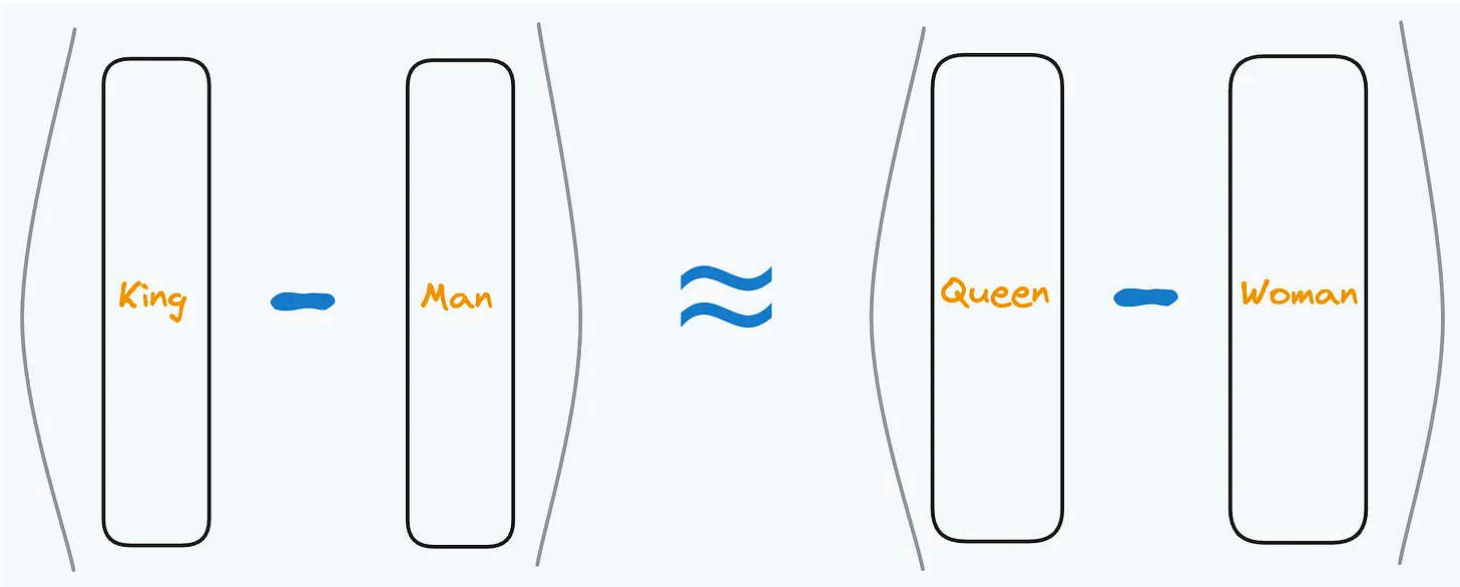
Consequently, other researchers would utilize those embeddings in their projects.

The most popular models at that time (around 2013-2017) were:

- Glove
- Word2Vec
- FastText, etc.

These embeddings genuinely showed some promising results in learning the relationships between words.

For instance, at that time, an experiment showed that the vector operation `(King - Man) + Woman` returned a vector near the word "Queen".



(King-Man) approximates to (Queen - Woman)

That's pretty interesting, isn't it?

In fact, the following relationships were also found to be true:

- `Paris - France + Italy ≈ Rome`
- `Summer - Hot + Cold ≈ Winter`
- `Actor - Man + Woman ≈ Actress`
- and more.

So while these embeddings did capture relative representations of words, there was a major limitation.

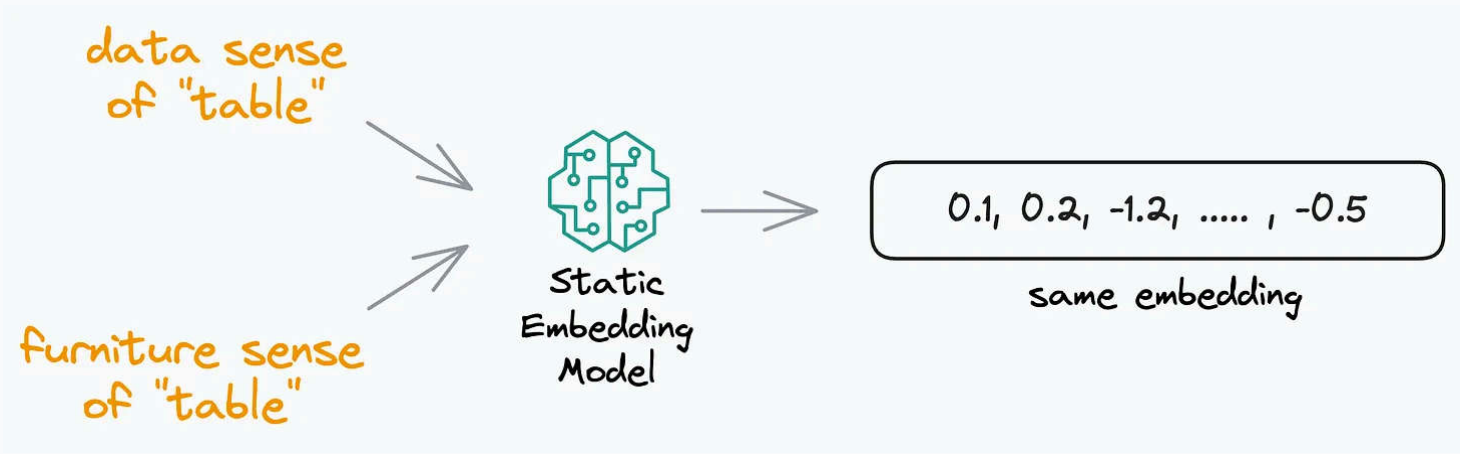
Consider the following two sentences:

- Convert this data into a **table** in Excel.
- Put this bottle on the **table**.

Here, the word "**table**" conveys two entirely different meanings:

- The first sentence refers to a "**data**" specific sense of the word "table".
- The second sentence refers to a "**furniture**" specific sense of the word "table".

Yet, static embedding models assigned them the same representation.



Same embedding for different usages of a word

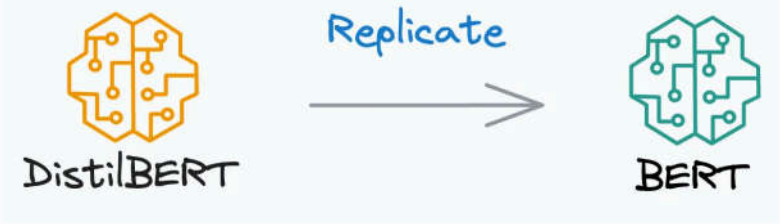
Thus, these embeddings didn't consider that a word may have different usages in different contexts.

But this was addressed in the Transformer era, which resulted in contextualized embeddings models powered by Transformers, such as:

- **BERT**: A language model trained using two techniques:

BERT pre-training

- - Masked Language Modeling (MLM): Predict a missing word in the sentence, given the surrounding words.
  - Next Sentence Prediction (NSP).
- **DistilBERT**: A simple, effective, and lighter version of BERT which is around 40% smaller:



Training DistilBERT

- - Utilizes a common machine learning strategy called student-teacher theory.
  - Here, the student is the distilled version of BERT, and the teacher is the original BERT model.
  - The student model is supposed to replicate the teacher model's behavior.
  - If you want to learn how this is implemented practically, we discussed it here: Model Compression: A Critical Step Towards Efficient Machine Learning.
- **ALBERT**: **A** Lite **BERT** (ALBERT). Uses a couple of optimization strategies to reduce the size of BERT:
  - Eliminates one-hot embeddings at the initial layer by projecting the words into a low-dimensional space.
  - Shares the weights across all the network segments of the Transformer model.

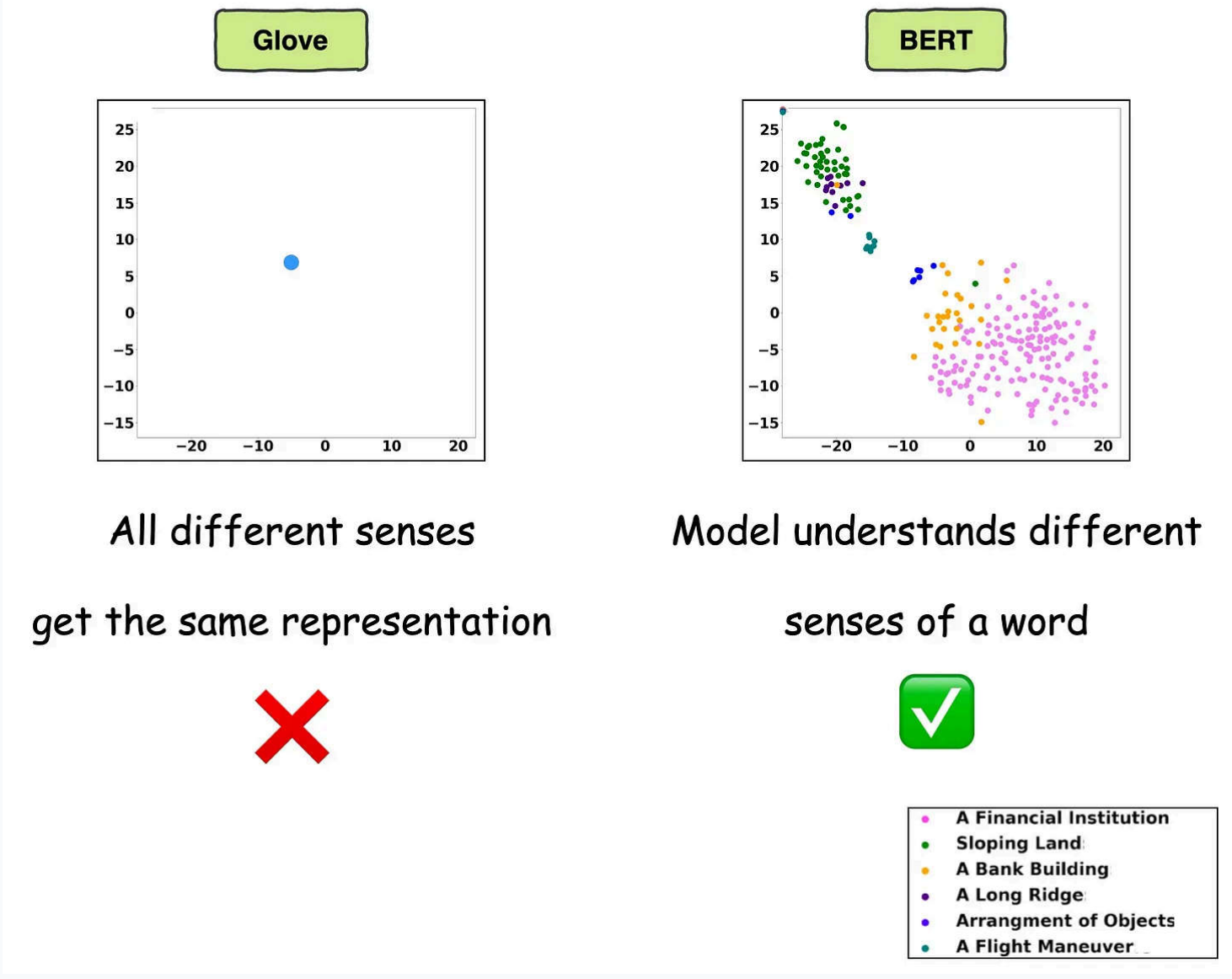
These models were capable of generating context-aware representations, thanks to their self-attention mechanism.

This would allow embedding models to dynamically generate embeddings for a word based on the context they were used in.

As a result, if a word would appear in a different context, the model would get a different representation.

This is precisely depicted in the image below for different uses of the word "Bank".

> For visualization purposes, the embeddings have been projected into 2d space using t-SNE.



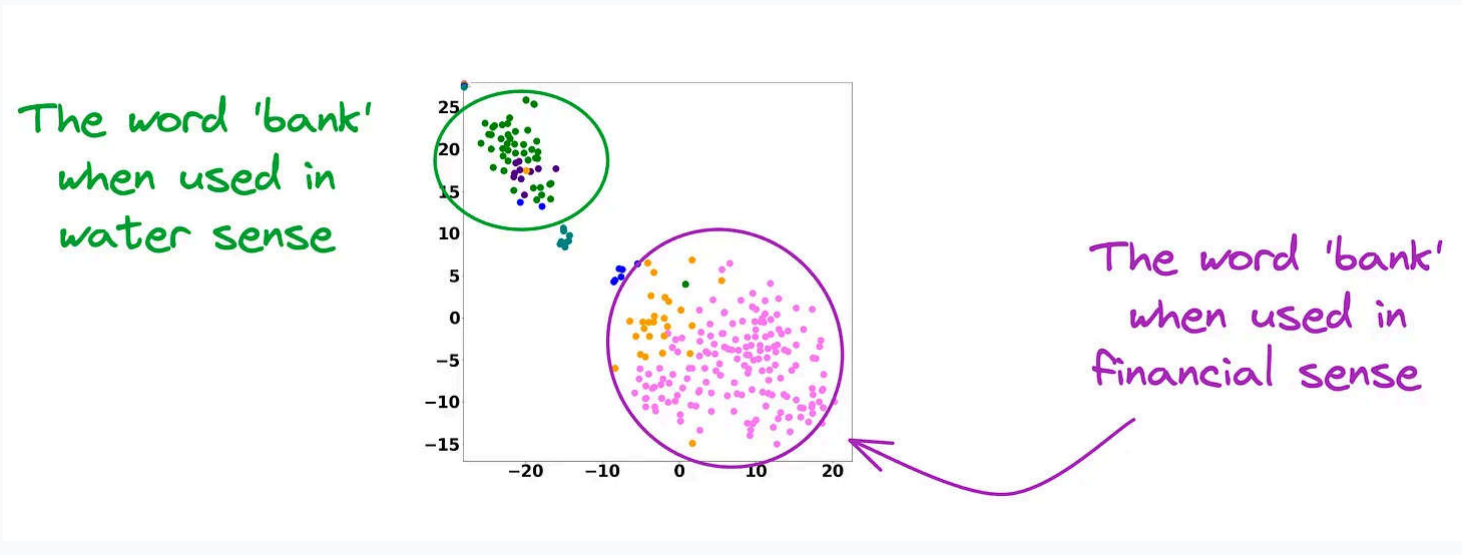
Glove vs. BERT on understanding different senses of a word

As depicted above, the static embedding models — Glove and Word2Vec produce the same embedding for different usages of a word.

However, contextualized embedding models don't.

In fact, contextualized embeddings understand the different meanings/senses of the word "Bank":

- A financial institution
- Sloping land
- A Long Ridge, and more.



As a result, they addressed the major limitations of static embedding models.

For those who wish to learn in more detail, I have published a couple of research papers on this topic:

- Interpretable Word Sense Disambiguation with Contextualized Embeddings.

- A Comparative Study of Transformers on Word Sense Disambiguation.

These papers discuss the strengths and limitations of many contextualized embedding models in detail.

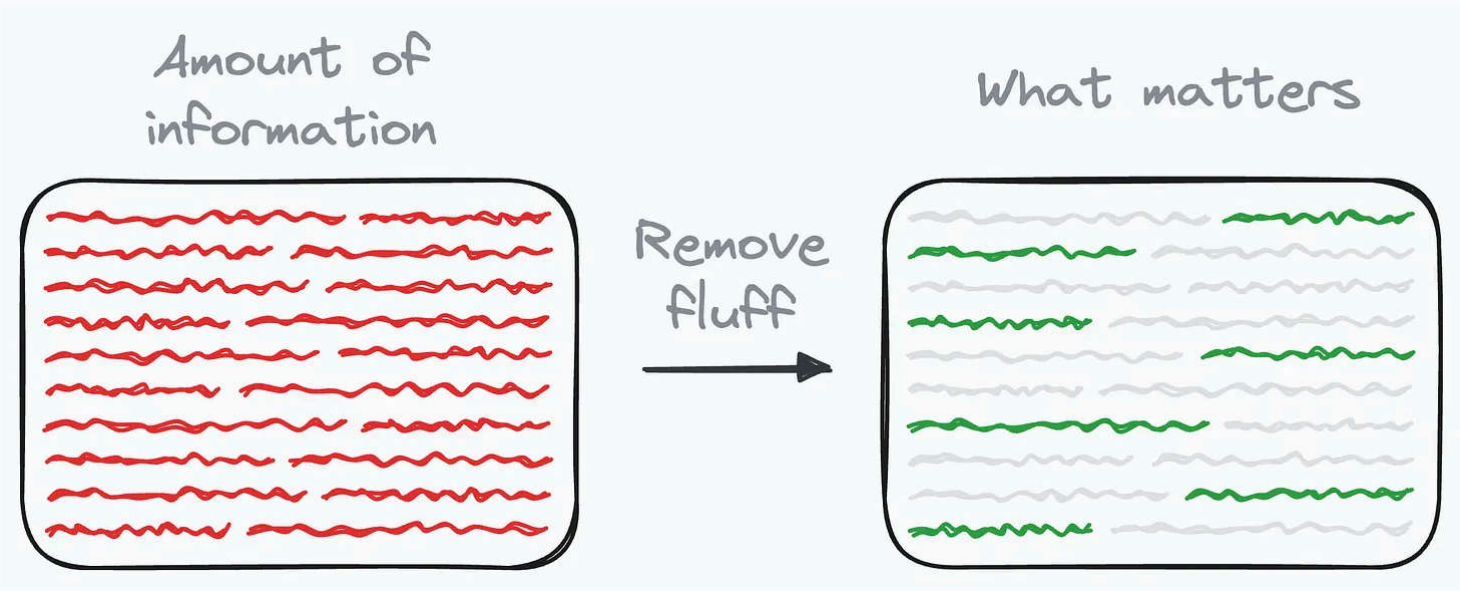
👉 Over to you: What do you think were some other pivotal moments in NLP research?

Thanks for reading Daily Dose of Data Science! Subscribe for free to learn something new and insightful about Python and Data Science every day. Also, get a Free Data Science PDF (550+ pages) with 320+ tips.

leila_weitzel@id.uff.br Subscribe

Are you overwhelmed with the amount of information in ML/DS?

Every week, I publish no-fluff deep dives on topics that truly matter to your skills for ML/DS roles.



For instance:

- A Crash Course of Model Calibration – Part 1
- Conformal Predictions: Build Confidence in Your ML Model's Predictions
- Quantization: Optimize ML Models to Run Them on Tiny Hardware
- A Crash Course on Causality – Part 1
- A Crash Course on Causality – Part 2
- A Beginner-friendly Introduction to Kolmogorov Arnold Networks (KANs)
- 5 Must-Know Ways to Test ML Models in Production (Implementation Included)
- A Beginner-Friendly Guide to Multi-GPU Model Training
- 8 Fatal (Yet Non-obvious) Pitfalls and Cautionary Measures in Data Science
- Implementing Parallelized CUDA Programs From Scratch Using CUDA Programming
- You Are Probably Building Inconsistent Classification Models Without Even Realizing
- And many many more.

Join below to unlock all full articles:

SPONSOR US

Get your product in front of 84,000 data scientists and other tech professionals.

Our newsletter puts your products and services directly in front of an audience that matters — thousands of leaders, senior data scientists, machine learning engineers, data analysts, etc., who have influence over significant tech decisions and big purchases.

To ensure your product reaches this influential audience, reserve your space here or reply to this email to ensure your product reaches this influential audience.

29 Likes · 4 Restacks

← Previous

Next →

Comments

Write a comment...

© 2024 Avi Chawla · Privacy · Terms · Collection notice
Substack is the home for great culture