



University of
Strathclyde
Humanities &
Social Sciences

Scientometrics in an Era of Large Language Models

Scott W. Cunningham

www.strath.ac.uk/humanities

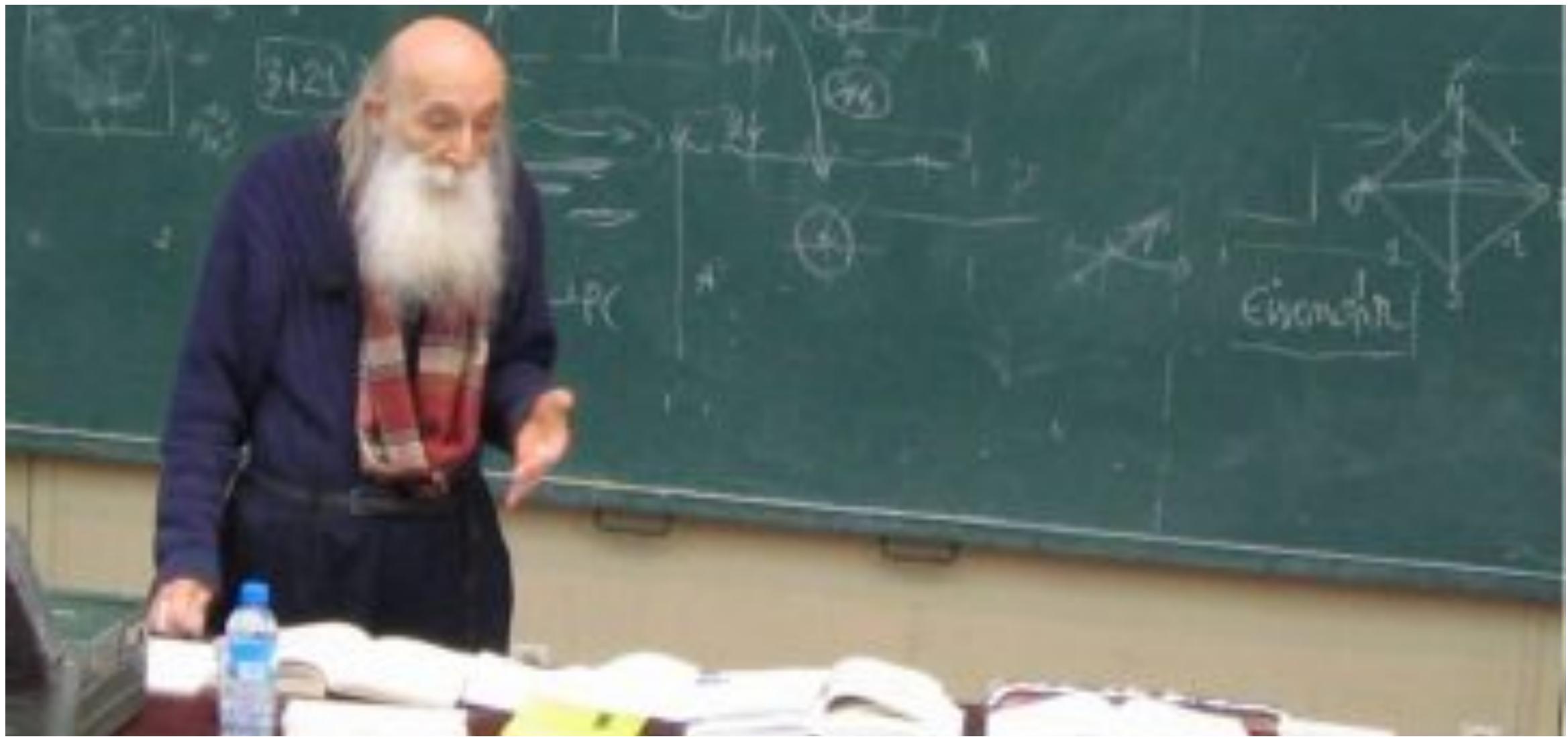
INTRODUCTION

- I'd like to talk about the emergence of large-language models including ChatGPT.
- I would like to convince you of their importance and impact by examining their current applications.
- Then I'd like to describe how and why I think the field of scientometrics will need to adapt to this new technology.

SUMMARY

- Brief demonstration
- Communicate an arc of progress
- Conclude with recommendations or opportunities

THE PAST



Jean-Paul Benzécri (1932 – 2019)

- Developed and popularised correspondence analysis
- Necessary technique for nominal data
- Displays and embeds of rows and columns
- Linguists found this technique invaluable for examining word usage patterns
- Precursor for science maps
- Parallel and later integrated with other techniques (PCA, SVD)

The AI Winter

- Noam Chomsky
- Influential linguist and cognitive scientist
- Argues for a “deep structure” to language
- Transformational rules
- Cognitive scientists attempted the same when encoding knowledge
- Automated translation promised
- Major funders became disenchanted

Information Retrieval

- Gerard Salton and Michael J. McGill
- *Introduction to Modern Information Retrieval* (1982)
- Pioneered the vector space model
- Queries and documents
- Scott Deerwester, et al.
- Bell Labs
- Latent Semantic Indexing
- Applies data reduction to vector spaces
- Eliminates noise, finds related words
- Early form of embedding

Deerwester, et al. (1983), p. 8

Unlike many typical uses of factor analysis, we are not necessarily interested in reducing the representation to a very low dimensionality, say two or three factors, because we are not interested in being able to visualize the space or understand it. But we do wish both to achieve sufficient power and to minimize the degree to which the space is distorted. We believe that the representation of conceptual space for any large document collection will require more than a handful of underlying independent "concepts", and thus that the number of orthogonal factors that will be needed is likely to be fairly large. Moreover, we believe that the model of a Euclidean space is at best a useful approximation. In reality, conceptual relations among terms and documents certainly involves more complex structures, including, for example, local hierarchies and non-linear interactions between meanings. More complex relations can often be made to approximately fit a dimensional representation by increasing the number of dimensions. In effect, different parts of the space will be used for different parts of the language or object domain. Thus we have reason to avoid both very low and extremely high numbers of dimensions. In between we are guided only by what appears to work best. What we mean by "works best" is not (as is customary in some other fields) what reproduces the greatest amount of variance in the original matrix, but what will give the best retrieval effectiveness.

Scientometrics



- Eugene Garfield (1925 - 2017)
- Comprehensive citation index
- Co-citation analysis
- Bibliometric coupling
- Main path analysis

Newer Methods

Topic models

- Including latent Dirichlet analysis (LDA) and non-negative matrix factorisation (NMF)
- Probabilistic or distributional assumptions on the data
- Unsequenced “bag of words”

Embedding models

- Word2Vec
- Stochastic Network Embedding (TSNE) (Van der Maaten and Hinton, 2012)

Problems and issues

- Lack of a standard vocabulary
- Lack of a reference implementation
- Necessary computation
- Incompressible language
- Non-linearity
- Visualising high dimensional spaces

THE PRESENT

Large Language Model Questions

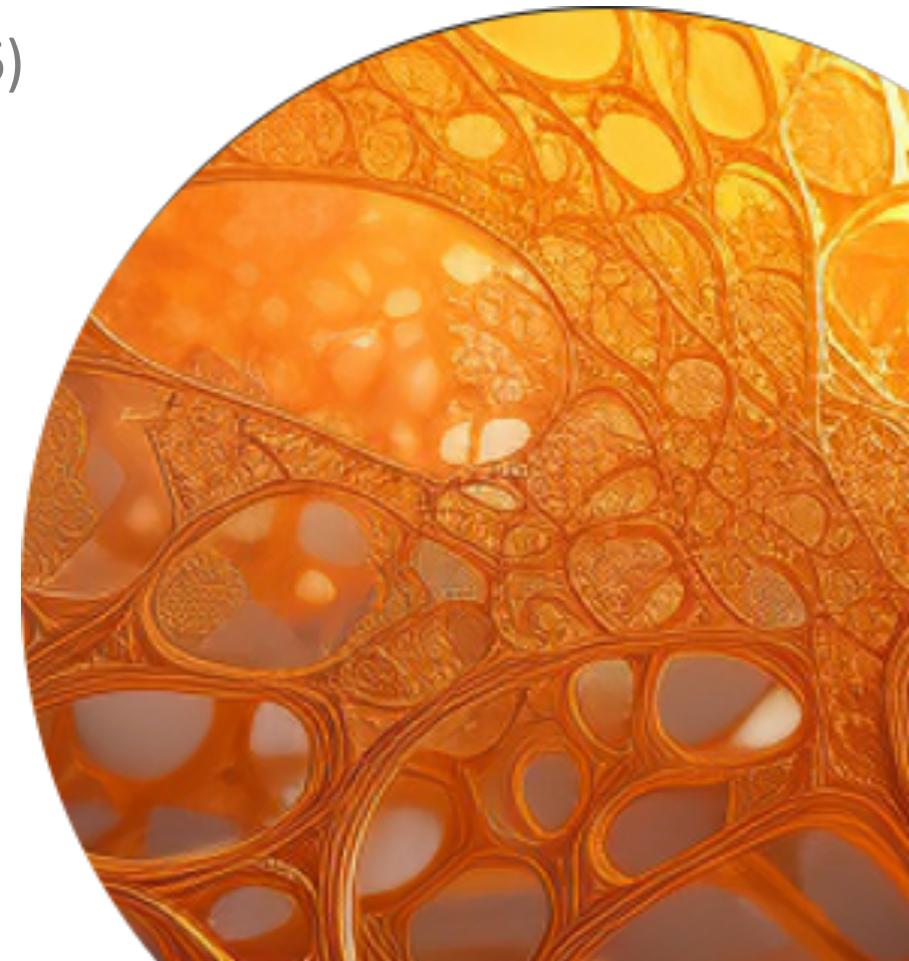
- What are they?
- How are they made?
- How are they used?
- How do they work?
- How can we best understand them?
- What do they do?

What are they?

- Large Language Models are neural networks
- Large quantities of data, deep structure, and clever input make them effective
- I'll briefly describe their architecture and transformation steps

New Advances

- The unreasonable effectiveness of data (Halevy et al. 2009)
- Deep learning (Hinton et al. 1986; Hinton et al. 2006)
- Recursion (c.f. Elman 1990)
- Attention is all you need (Vaswani et al. 2017)



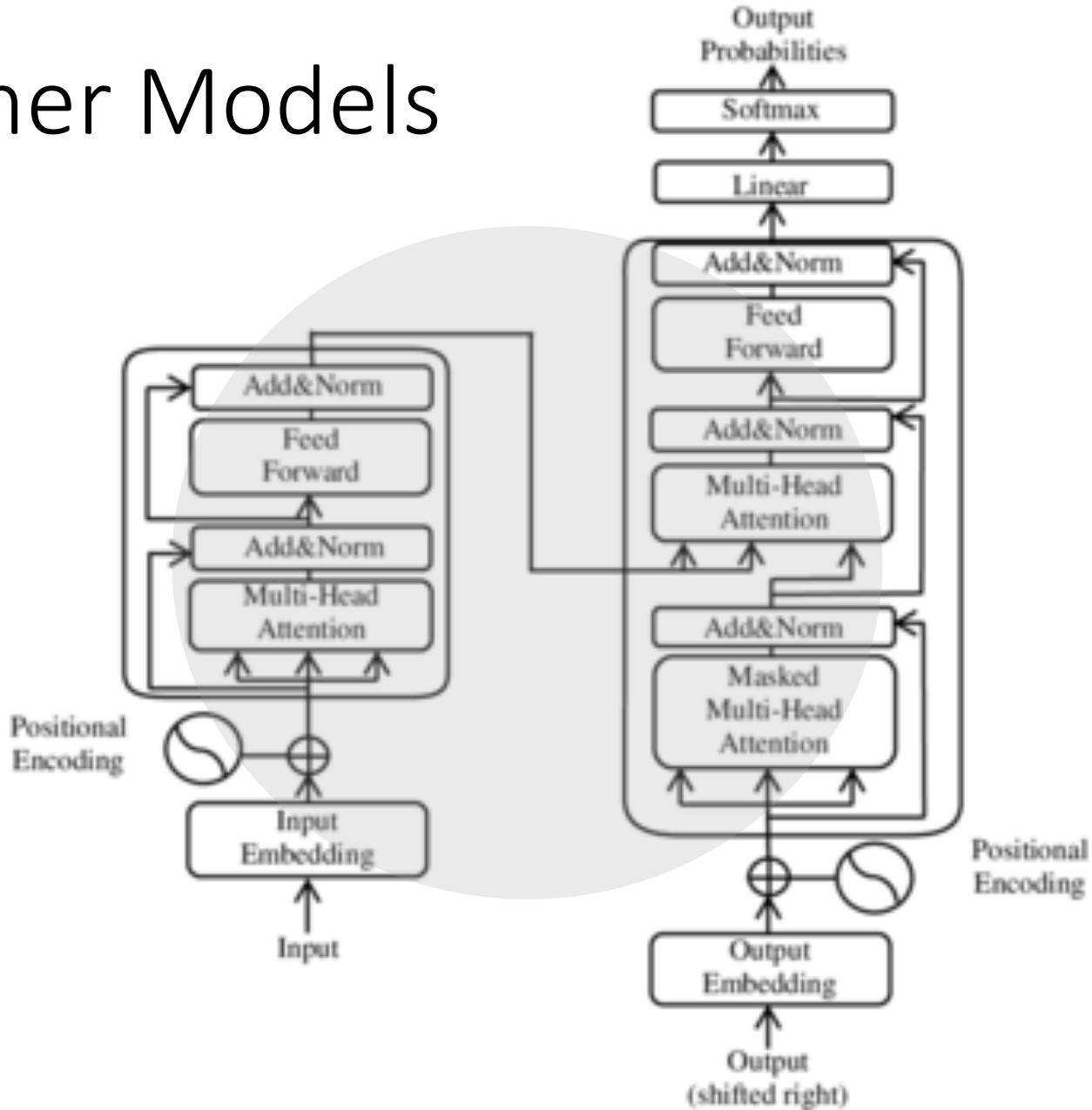
Transformer Models

Encoding
BERT

*Bidirectional
Encoder
Representation*

Decoding
GPT

*Generative
Pretrained
Transformer*



How are they made?

Somewhere in these architectural choices performance improvements are made

- Tokenisation and embedding
- Positions and sequences
- Normalisation and activation
- Parameters and layers
- Data and training

How are they used?



How can we best understand them?



“A blurry JPEG of all the text on the Web.”

“Think of ChatGPT as a blurry JPEG of all the text on the Web. It retains much of the information on the Web, in the same way that a JPEG retains much of the information of a higher-resolution image, but, if you’re looking for an exact sequence of bits, you won’t find it’ all you will ever get is an approximation.”

“Indeed, a useful criterion for gauging a large language model’s quality might be the willingness of a company to use the text that it generates as training material for a new model.”

“Sometimes it’s only in the process of writing that you discover your original ideas. . . Your first draft isn’t an unoriginal idea expressed clearly; it’s an original idea expressed poorly.”

Chiang (2023)

“Behold the shoggoth”



Henry Farrell @henryfarrell · 4d ...
1. [economist.com/by-invitation/...](https://economist.com/by-invitation/) So Cosma Shalizi and I have a piece in the Economist on shoggoths. The Large Language Models driving ChatGPT and Bing are indeed Lovecraftian monsters. But so too are markets, bureaucracies, and even democracy, if squinted at askance.



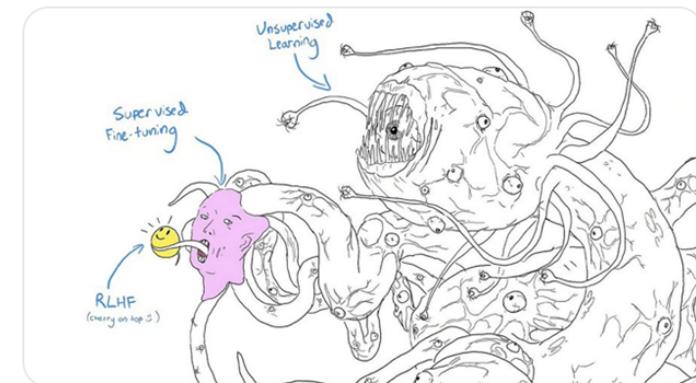
Henry Farrell @henryfarrell · 4d ...
2. The idea is straightforward. Like [@AlisonGopnik](#) we don't see LLMs as analogous to individual human intelligences. Instead, they're "cultural technologies" - systems for summarizing human created knowledge and making it useful. This puts the shoggoth meme in a different context

2 14 68 9,343 ⬆



Henry Farrell @henryfarrell · 4d ...
Replies to [@henryfarrell](#)
5. The point is that this is _nothing new._ Here, we're riffing on a fantastic piece that Cosma wrote a decade ago, arguing that markets and bureaucracies were vast, inhuman distributed systems of information-processing "the coldest of all cold monsters" bactra.org/weblog/699.html

3 8 68 5,109 ⬆



14 210 505 166K ⬆

(Farrell and Shalizi, 2023)

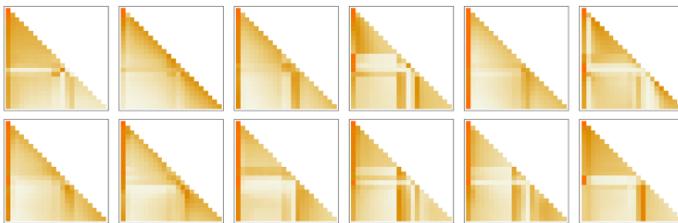
“We want to believe”



(art by Alexis Poles,
referenced by Baussells
2015)

How do they work?

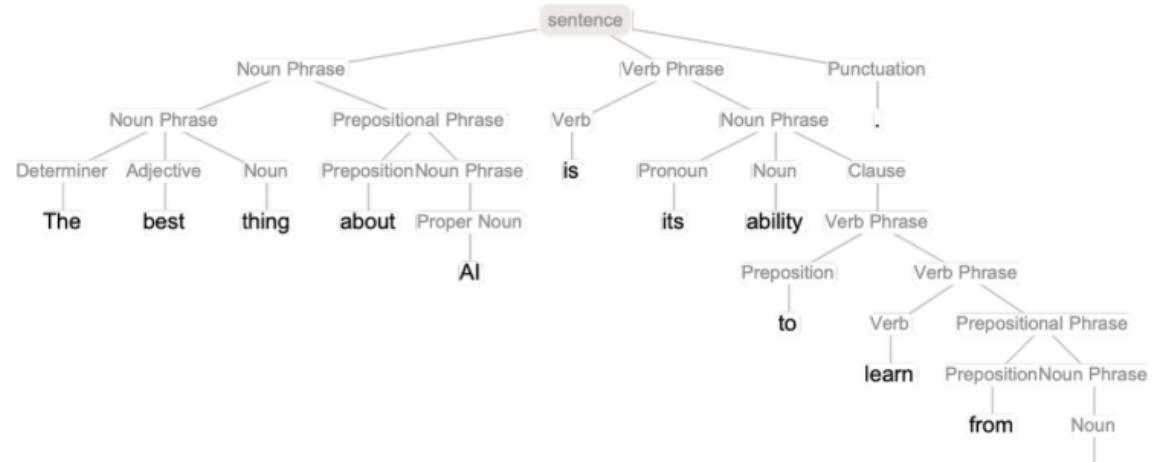
Attention Heads



Layers



Parse Trees

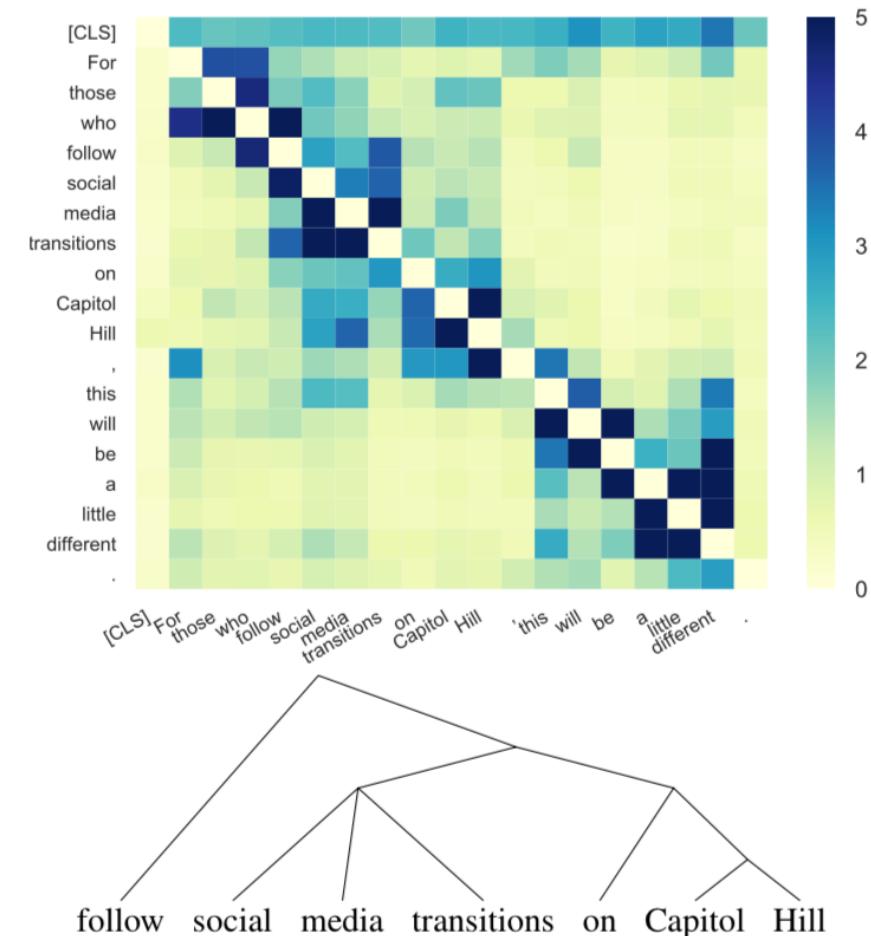


Wolfram (2023)

Probing Classifiers

A Primer in BERTology (Rogers et al. 2020)

- Probing Classifiers (Belinkov 2021)
- Fill-in-the-gap Probes
- Analysis of Self-Attention Weights



What do they do?

- Collecting data
- Co-teaming with OpenAI
- Implementation
- Pre-processing
- Analysis
- Visualisation

Collecting Data

Search > Results for technological fo... > Results for technological forecasting and social change (Publication Titles) ...

284 results from Web of Science Core Collection for:

Q technological forecasting and social change (Publication Titles)

Refined By: Publication Years: 2023 X Clear all

GD Copy query link

Publications You may also like...

Refine results

Search within results... Q

Filter by Marked List

Quick Filters

- Review Article 14
- Open Access 86
- Associated Data 1

Citation Topics Meso

- 6.3 Management 66
- 6.115 Sustainability Science 18
- 6.185 Communication 17
- 6.10 Economics 12
- 6.294 Operations Research & Management... 11

See all >

0/284 Add To Marked List Export ▾

1 Small steps for the big hit: A dynamic capabilities perspective on business networks and non-disruptive digital technologies in SMEs

Soluk, J.; Decker-Lange, C. and Hack, A.

Jun 2023 | TECHNOLOGICAL FORECASTING AND SOCIAL CHANGE 191

This study explores how small and medium-sized enterprises (SMEs) build on business networks to develop dynamic capabilities that ultimately foster the adoption of non-disruptive digital technologies (DT). Multiple mediation analysis is used to test this relationship, drawing on cross-industry secondary data from 1470 British SMEs. The results reveal that SMEs rely on business networks to support the parallel effects of three dynamic capabilities (HR, strategic planning, and marketing capabilities). However, the results show that SMEs with higher levels of HR and strategic planning tend to have more dynamic capabilities, which in turn lead to higher levels of DT adoption. The findings suggest that SMEs can benefit from building strong business networks to support their digital transformation journey.

Find it at Strathclyde Free Full Text From Publisher ...

2 A socio-technical transition perspective on positive tipping points in climate change mitigation debates: The role of interacting feedback loops in offshore wind and electric vehicles acceleration

Geels, F.W. and Ayoub, M.

Aug 2023 | May 2023 (Early Access) | TECHNOLOGICAL FORECASTING AND SOCIAL CHANGE 193

This paper engages with climate mitigation debates on positive tipping points, which attract increasing attention. It argues that these debates are often framed in either technological or social tipping point approaches. Building on recent attempts to overcome this dichotomy, the paper introduces a socio-technical transitions perspective which shows how co-evolutionary interactions between techno-economic im-

PT	AU	BA	BE	GP	AF	BF	CA	TI	SO	SE	BS
Z9	LA	DT	CT	CY	CL	SP	HO	DE	ID	AB	C1
PD	C3	RP	EM	RI	OI	FU	FP	FX	CR	NR	TC
DI	U1	U2	PU	PI	PA	SN	EI	BN	J9	JI	
HC	PY	VL	IS	PN	SU	SI	MA	BP	EP	AR	
HC	DL	D2	EA	PG	WC	WE	SC	GA	PM	OA	
J	HP	DA	UT								

Soluk, J.; Decker-Lange, C.; Hack, A.
Soluk, Jonas; Decker-Lange, Carolin; Hack, Andreas
Small steps for the big hit: A dynamic capabilities perspective on business networks and non-disruptive digital technologies in SMEs TECHNOLOGICAL FORECASTING AND SOCIAL CHANGE English Article
Digital transformation; Digital technology adoption; Non-disruptive digital technologies; Digital innovation;

WOS:000974581800001 Small steps for the big hit: A dynamic capabilities perspective on business networks and non-disruptive digital technologies in SMEs This study explores how small and medium-sized enterprises (SMEs) build on business networks to develop dynamic capabilities that ultimately foster the adoption of non-disruptive digital technologies (DT). Multiple mediation analysis is used to test this relationship, drawing on cross-industry secondary data from 1470 British SMEs. The results reveal that SMEs rely on business networks to support the parallel effects of three dynamic capabilities (HR, strategic planning, and marketing capabilities). However, the results show that SMEs with higher levels of HR and strategic planning tend to have more dynamic capabilities, which in turn lead to higher levels of DT adoption. The findings suggest that SMEs can benefit from building strong business networks to support their digital transformation journey.

Co-Teaming with OpenAI

sc How can I preprocess text for use in BERT with the python language?

To preprocess text for use with BERT in Python, you can follow these steps:

1. Install the required libraries:

```
pip install transformers
```
2. Import the necessary modules:

```
python
from transformers import BertTokenizer
import torch
```
3. Load the pre-trained BERT tokenizer:

```
python
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
```
4. Tokenize the text:

```
python
```

Regenerate response

Copy code

sc How can I embed preprocessed text using BERT and transformers in Python?

To embed preprocessed text using BERT and transformers in Python, you can follow these steps:

1. Install the required libraries: Begin by installing the necessary libraries. You can use pip to install the `transformers` library, which provides an easy interface for using BERT and other transformer models. Run the following command in your terminal or command prompt:

```
pip install transformers
```
2. Import the required modules: In your Python script or notebook, import the required modules from the `transformers` library:

```
python
from transformers import BertTokenizer, BertModel
import torch
```
3. Load the pre-trained BERT model and tokenizer: Choose a pre-trained BERT model that

Implementation

This screenshot shows a Jupyter Notebook interface titled "jupyter Data Handling". The notebook has two cells:

```
In [3]: with open(file_path, 'w') as file:  
    for sublist in data:  
        file.write(line + '\n')  
  
# Example usage  
data = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]  
file_path = 'output.txt'  
print_list_of_lists_to_file(data, file_path)  
  
In [34]: import csv  
file_path = 'D:/TFSC2023.txt'  
def import_tab_separated_file(file_path):  
    data = []  
    header = []  
    with open(file_path, 'r', encoding='utf-8-sig') as file:  
        reader = csv.reader(file, delimiter='\t')  
        for n, row in enumerate(reader):  
            if (n > 0):  
                data.append(row)  
            else:  
                header.append(row)  
    return header, data  
  
# Usage example  
import time  
header, tab_data = import_tab_separated_file(file_path)  
  
for n, el in enumerate(header):  
    print(n, el)  
  
for n, el in enumerate(tab_data):  
    words = el[2].split(' ')  
    print(n, len(words))
```

The output of the second cell is:

```
0 PT  
1 AU  
2 BA  
3 BE
```

This screenshot shows a Jupyter Notebook interface titled "jupyter BiblioBERT". The notebook has several cells, with the first few showing the download of a pre-trained BERT tokenizer:

```
In [3]: from transformers import BertTokenizer  
import torch  
  
In [4]: tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')  
  
Downloading (...)solve/main/vocab.txt: 232k/? [00:00<00:00, 3.14MB/s]  
C:\Users\seb19206\Anaconda3\lib\site-packages\huggingface_hub\file_download.py:133: UserWarning:  
'huggingface_hub' cache-system uses symlinks by default to efficiently store duplicated files but  
your machine does not support them in C:\Users\seb19206\.cache\huggingface\hub. Caching files wil  
l still work but in a degraded version that might require more space on your disk. This warning c  
an be disabled by setting the 'HF_HUB_DISABLE_SYMLINKS_WARNING' environment variable. For more de  
tails, see https://huggingface.co/docs/huggingface\_hub/how-to-cache#limitations.  
To support symlinks on windows, you either need to activate Developer Mode or to run Python as an  
administrator. In order to see activate developer mode, see this article: https://docs.microsoft.com/en-us/windows/apps/get-started/enable-your-device-for-development  
warnings.warn(message)  
  
Downloading (...)okenizer_config.json: 28.0/28.0 [00:00<00:00,  
100% 780B/s]  
  
Downloading (...)live/main/config.json: 570/570 [00:00<00:00,  
100% 13.7kB/s]
```

Subsequent cells show the tokenization process:

```
In [5]: text = "Your input text goes here."  
tokens = tokenizer.tokenize(text)  
  
In [6]: input_ids = tokenizer.convert_tokens_to_ids(tokens)  
  
In [7]: input_ids = [tokenizer.cls_token_id] + input_ids + [tokenizer.sep_token_id]  
  
In [8]: attention_masks = [1] * len(input_ids)  
  
In [9]: max_length = 128  
padding_length = max_length - len(input_ids)
```

Preprocessing

```
▶ text = "Your input text goes here."
tokens = tokenizer.tokenize(text)

▶ input_ids = tokenizer.convert_tokens_to_ids(tokens)

▶ input_ids = [tokenizer.cls_token_id] + input_ids + [tokenizer.sep_token_id]

▶ attention_masks = [1] * len(input_ids)

▶ max_length = 128
padding_length = max_length - len(input_ids)

if padding_length > 0:
    input_ids = input_ids + ([tokenizer.pad_token_id] * padding_length)
    attention_masks = attention_masks + ([0] * padding_length)
else:
    input_ids = input_ids[:max_length]
    attention_masks = attention_masks[:max_length]

▶ input_ids = torch.tensor(input_ids)
attention_masks = torch.tensor(attention_masks)
```

```
print(input_ids)

tensor([[ 101,  3617,  2740,  7248,  2005,  1996,  9750,  1029,  3145,  9886,
         1998,  8192, 13500,  1998,  3971,  2000,  4769,  2068,  3617,  9871,
        7248,  1006, 28144,  4523,  1007,  5050,  1037,  4659,  2047,  9575,
       2008,  2071,  3073,  1037,  7070, 13711,  2000,  3558,  3078,  2729,
      1011,  2005,  2742,  1010,  2011,  8161,  5366,  1010,  9229,  3229,
     2000,  9871,  1010,  1998,  4352,  5776,  8822,  1012,  2174,  1010,
    2107,  7248,  2024,  3701,  2109,  2651,  2011,  1996,  3920,  8213,
   1010,  2029,  9005,  1037,  1000,  3617, 11443,  1000,  2090,  1996,
  3920,  1998,  1996,  9750,  1012,  2023,  3720,  8704,  2000,  6709,
  1024,  1045,  1007,  1996,  8690,  3145, 13500,  2008,  26402,  9886,
  1998,  8192,  1997, 28144,  4523,  2011,  1996,  9750,  1010,  1998,
  2462,  1007,  2054, 28144,  2361, 11670,  2064,  2079,  2000, 10956,
  3445,  9886,  1998,  8192,  2011,  1996,  9750,  1012,  1996,  3720,
  9891,  2006, 24209, 11475, 27453,  7636,  2007,  9750,  1998, 21053,
  2832,  2951,  2013,  1037,  2350,  4467, 28144,  2361,  1012,  2057,
  2424,  2008,  1996,  9750,  23084,  2015,  2048,  3145, 13500,  2000,
  3988,  9886,  1997, 28144,  4523,  1024,  1045,  1007,  4997, 13818,
  1998,  2974, 10089,  1998,  2462,  1007,  2028,  3145,  8803, 12473,
  2119,  9886,  1998,  8192,  1011,  3768,  1997,  3404,  1012,  1996,
  4106,  2036, 14847,  3674,  2458, 15690,  2005,  28144,  2361,  7620,
  2000,  2488,  8752,  1996,  3791,  1997,  1996,  9750,  1010,  2164,
  15690,  2005,  4646,  2458,  1998, 21727,  2495,  3450,  1012,  2057,
  3073,  2019,  6377,  7705,  2041, 16992,  1996,  3145, 13500,  8690,
  1998,  3971,  2000,  4769,  2068,  1012,  1999,  2061,  2725,  1010,
  2057,  9002,  2000,  1996,  3906,  2006,  1049,  20192, 24658,  1998,
  2000,  1996,  3906,  2006,  7248,  1999,  9871,  1012,  102,  0,
          0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
          0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
```

Analysis

```
from transformers import BertTokenizer, BertModel
import torch

model_name = 'bert-base-uncased'
tokenizer = BertTokenizer.from_pretrained(model_name)
model = BertModel.from_pretrained(model_name)

Downloading model.safetensors: 100%
```

```
4
```

```
▶ print(embeddings)
tensor([[[ -0.9762, -0.0995, -0.8006, ..., 0.0029, 1.0137, -0.4809],
        [-0.7192, 1.2761, -0.5353, ..., 0.2435, 0.5061, 1.1344],
        [ 0.0832, 0.5186, -0.7893, ..., -0.3839, 0.0572, 0.4023],
        ...,
        [ 0.5694, -0.1146, 0.4969, ..., -0.8004, -0.2793, -1.2694],
        [ 0.4882, -0.2114, 0.4366, ..., -0.7421, -0.2004, -1.2331],
        [ 0.4308, -0.0469, 0.3925, ..., -0.7531, -0.2243, -1.2997]]])
```

```
▶ # there are 250 tokens, each in a dimension of 768
embeddings.shape
3]: torch.Size([1, 250, 768])
```

```
▶ import numpy as np
X = embeddings[0].numpy()
```

```
▶ print(X.shape)
(250, 768)
```

```
▶ np.sum(X, axis=0).shape
7]: (768,)
```

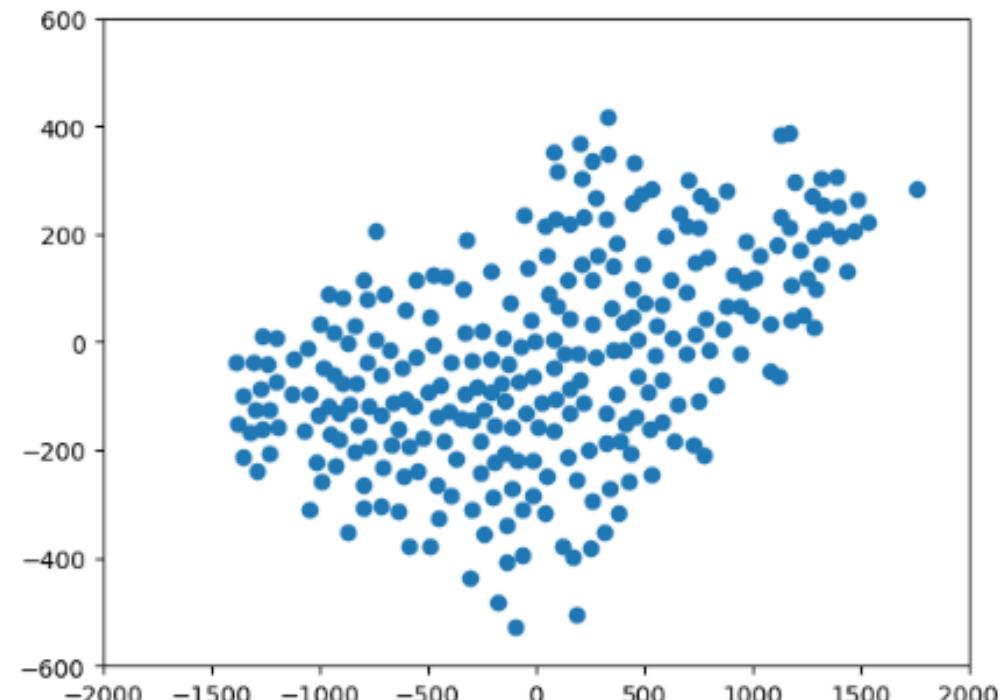
Visualisation

```
# Perform t-SNE embedding
tsne = TSNE(n_components=2, random_state=42, init='pca')
embedded_data = tsne.fit_transform(data)
```

```
x = embedded_data[:,0]
y = embedded_data[:,1]
embedded_data.shape
```

```
(283, 2)
```

```
import matplotlib.pyplot as plt
plt.axis([-2000, 2000, -600, 600])
plt.scatter(x,y)
<matplotlib.collections.PathCollection at 0x1dca717a9a0>
```



THE FUTURE

Challenges for Scientometrics

- Scientific language is not just natural language
- We need to represent more than words (authors, institutions, references, citations, patents)
- Attention windows are too short (about 400 words)
- Scientific knowledge is embodied in communities, over time
- We need to do more than chat with LLM outputs
- Most of us can't afford the compute to calculate our own models
- Correctness
- Reasoning

Opportunities

- There is an active community experimenting with what works
- The parameters of these models have leaked and are readily available
- Fundamental and hybrid algorithms
- Models of scientific text (SciBERT, SciSciNet) (Beltagy, et al. 2019; Lin et al. 2023)
- The model has language capabilities, and it transports
- Probing the networks for science (Rogers et al. 2020)
- Distilling the networks (Hinton et al. 2014)

Uptake

- Quite limited
- Most of the work applies bibliometrics to machine learning
- But see Sachini et. al (2022) in a recent publication in *Quantitative Science Studies* using BERT for classification

The screenshot shows a search interface for the Web of Science Core Collection. At the top left is a black sidebar with icons for 'SEARCH' (magnifying glass), 'MENU' (three horizontal lines), 'FOLDER' (file folder), and 'RELOAD' (refresh symbol). The main search bar contains the query: 'BERT or transformer or ChatGPT (Topic) and scientometrics or bibliometrics or informetrics (Topic)'. Below the search bar, a message says '10 results from Web of Science Core Collection for:' followed by the search terms. To the right of the search bar are three purple buttons: 'Analyze Results', 'Citation Report', and 'Create Alert' (with a bell icon). At the bottom of the search bar area, there is a suggestion: 'Did you mean? BERT or transformer or ChatGPT (Topic) and scientometrics or bibliometrics or informetrics (Topic) | 10 results'

Continuing Need

- We still need scientometrics
- “Science of scientific knowledge”
- Applied purposes as well
 - Communicating
 - Tracking new knowledge
 - Technological problem solving (Swanson 1986; Smalheiser and Swanson, 1998)
 - Building research ecologies and districts

RECOMMENDATIONS

- We need to move to open-source computing languages including Python and R.
- The scientometric community needs to take advantage of the large computational investments going on in LLM, including BERT.
- We should take serious notice of the architectures behind these models.
- The availability of these models should stimulate new sorts of scientometric questions, and new applications for the policy and management communities.

CONCLUSIONS

- I've presented a reflexive account of LLM
- I hope to have contributed by making LLM an object of study, and also by advocating the use LLM to facilitate our own studies
- I think that LLM have solved the long-standing problem of effective representations of text
- You've permitted me an opportunity to experiment with ChatGPT in the making of this presentation

Appendix: Extra Slides

Tokenisation and Embedding

Tokenisation

- Stop words and stemming
- Byte Pair Encoding
- SentencePiece library

Embedding

- Embedding of words in high-dimensional spaces
- Spaces encode meaning
- Word2Vec was one of the leading

Position and Sequence

- Position
 - The order of the word in the sentence is important to meaning
 - Grammar and syntax; other poorly understood reasons
 - We need to encode this
- Sequence
 - Addressing by repeated parsing or scanning
 - Recurrent Neural Networks
 - Learn the words which matter
 - Attention

Normalisation and Activation

Normalisation

- The layers are constantly being rescaled back to a unit length

Activation

- The layers use a non-linear transformation
- This enables a more expressive representation
- Rectified Linear Unit (ReLU)
- Returns non-negative values

Parameters and Layers

- A small ChatGPT model contains 12,873,072 parameters
- Attention heads [A] 4
- Embedding size [E] 256
- Layers [L] 4
- Sequence Length [S] 512
- Vocabulary [V] 30,000
- Token embedding layers
[V x E]
- Position embedding layer
[S x E]
- Transformer blocks
[A x L x E x E]

Data and Training

Data

- Web pages
- Books
- Scientific publications
- Social media
- Open-domain question-answering data sets
- Educational resources
- User-provided data
- One trillion words

Training

- Models are batch trained and so are static
- They are periodically retrained
- Applications fine tune the end results of the model
- Researchers estimate that training GPT-3 cost OpenAI around \$5 million
- Objective functions (predicting missing words, predicting sentences)

Realistic

- Question-answering systems:
These models can be deployed in systems that provide answer to relevant questions based on context and available information.
- It can assist in these tasks by offering guidance, answering questions about proper procedures, providing safety instructions, troubleshoot, and assisting with training manuals.



Investigative

- Code generation: Some ChatGPT models are tailored to understand programming syntax and can be used to generate or suggest code snippets.
- Sentiment analysis: ChatGPT models can interpret and analyze various sentiments and emotions within texts, providing valuable insights for businesses and researchers.



Microsoft



Artistic

- Gaming and storytelling: ChatGPT models enhance the depth and variety of in-game characters, allowing them to engage in more natural and responsive conversations.
- Content generation: These models help create articles, blog posts, social media content, and more by generating relevant and coherent text.
- Creative writing: ChatGPT models can assist aspiring writers by suggesting ideas, generating plotlines, or even creating dialogues between characters.



Social

- Customer service: ChatGPT models are used to create chatbots for handling customer queries, improving response time and efficiency, and reducing the workload on human agents.
- Education and tutoring: These models can assist in education systems by generating learning material, answering questions, and offering personalized instruction.



INTERCOM

Enterprising

- Business Development: ChatGPT can help develop business plans, marketing strategies, and sales pitches. It can provide general knowledge about regulations, financial matters, or industry trends, though it doesn't replace the expertise of trained professionals in financial or legal fields.



Conventional

- Language translation: These models assist in language translation tasks by understanding the context and generating accurate translations.
- Virtual assistants: ChatGPT models can be integrated into personal virtual assistants to provide smart, context-aware information, reminders, and other assistance.



nreal