**JAECS 49th Conference**
第49回大会（30周年記念大会）

# Corpus AI: Integrating Large Language Models (LLMs) into a Corpus Analysis Toolkit

**Laurence Anthony**
Professor
Faculty of Science and Engineering
Waseda University, Tokyo, Japan
anthony@waseda.jp
http://www.laurenceanthony.net/

Kansai University, Osaka, Japan.
September 9-10, 2023 (Sat.-Sun.). https://jaecs2023.sakura.ne.jp/WP/

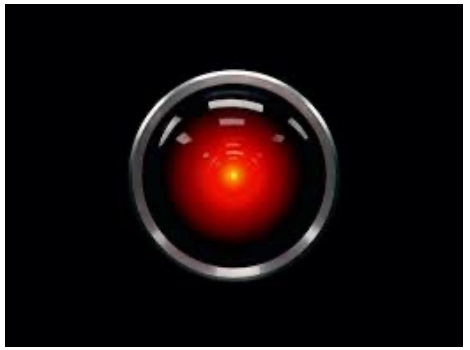
# Overview

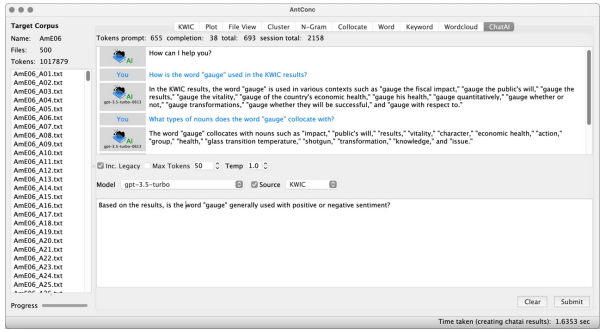
- Brief introduction to Large Language Models (LLMs)
  - definition, basic design, key components
  - main uses (and emergent properties)
- Challenges in using LLMs for corpus-based research
  - limitations, concerns, and dangers of LLMs
  - practical difficulties
- Integrating LLMs into a corpus analysis toolkit (AntConc)
  - strategy and design
  - live demonstration
  - next steps

OpenAI

Meta AI

Google AI





# Brief introduction to Large Language Models (LLMs)

definition; basic design; key components; main uses (and emergent properties)

# Brief introduction to Large Language Models (LLMs)

## Definition; Basic design; Key components

- Definition
  - Large Language Models (LLMs) are a deep learning architecture for performing natural language processing (NLP) tasks.
  - LLMs are neural networks (usually built on a transformer model architecture)
- Basic design and construction
  - Encoder
    - tokenizes the data
    - finds (learns) relationships between the tokens
  - Decoder
    - predicts the next token in a sequence
    - generates output text

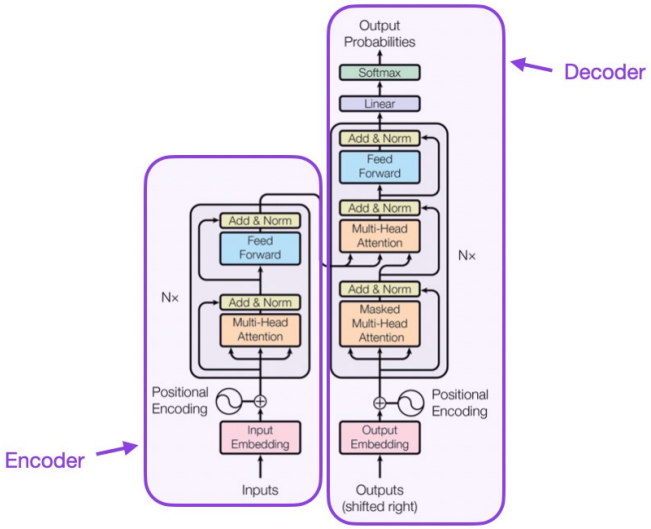


Figure 1: The Transformer - model architecture.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

# Brief introduction to Large Language Models (LLMs)

## Definition; Basic design; Key components

- Key components
  - Embedding Layer
  - Transformer Architecture
  - Attention Mechanisms
  - Positional Encoding
  - Multi-Head Attention
  - Feedforward Neural Networks
  - Layer Normalization and Residual Connections
  - Output Layer
  - Vocabulary and Tokenizer
  - Pre-training and Fine-tuning
  - Attention Masking

Large Language Models (LLMs) typically consist of several key components that work together to understand and generate human language effectively. The primary components of an LLM include:

1. **Embedding Layer:** This layer converts input tokens (e.g., words or subwords) into numerical representations, often called word embeddings or token embeddings. These embeddings capture the semantic meaning of words and their context in the language.
2. **Transformer Architecture:** The core architecture of most modern LLMs is the Transformer. It includes multiple layers of self-attention mechanisms and feedforward neural networks. Transformers enable the model to capture long-range dependencies and relationships between words in a sequence.
3. **Attention Mechanisms:** Within the Transformer architecture, attention mechanisms allow the model to assign different levels of importance (attention weights) to different parts of the input sequence when processing a given token. Self-attention is a fundamental part of understanding context in LLMs.
4. **Positional Encoding:** Since the Transformer doesn't have built-in notions of word order or position, positional encoding is added to the input embeddings to provide information about the position of tokens in the sequence. This helps the model understand word order.
5. **Multi-Head Attention:** LLMs often use multi-head attention mechanisms, where attention is computed in parallel multiple times with different sets of learnable parameters. This allows the model to attend to different aspects of context simultaneously.
6. **Feedforward Neural Networks:** After attention mechanisms, each token's representation is processed through feedforward neural networks, which include activation functions and weight matrices to transform the input.
7. **Layer Normalization and Residual Connections:** These are techniques used to stabilize training in deep neural networks. Layer normalization normalizes activations within each layer, and residual connections help prevent the vanishing gradient problem.
8. **Output Layer:** The final layer of the LLM maps the internal representations back into the vocabulary space, producing a probability distribution over possible output tokens. This allows the model to generate text or make predictions.
9. **Vocabulary and Tokenizer:** LLMs have a predefined vocabulary of words or subwords. A tokenizer is used to break text into tokens that are part of the vocabulary. Out-of-vocabulary words may be handled using subword tokenization.
10. **Pre-training and Fine-tuning:** LLMs are pre-trained on massive text corpora to learn language patterns and knowledge. After pre-training, they can be fine-tuned on specific tasks, such as text classification, translation, or question-answering.
11. **Attention Masking:** In certain applications, such as text generation, an attention mask may be used to prevent the model from attending to future tokens or certain positions in the input.

# Brief introduction to Large Language Models (LLMs)

## Definition; Basic design; Key components

- Key components
  - Embedding layer
    - to covert input tokens into numerical representations
  - Feedforward neural network layer (FFN)
    - to capture long-range dependencies and relationships between words
  - Attention mechanism
    - to assign weights to different parts of the input sequence
  - …

Large Language Models (LLMs) typically consist of several key components that work together to understand and generate human language effectively. The primary components of an LLM include:

1. **Embedding Layer:** This layer converts input tokens (e.g., words or subwords) into numerical representations, often called word embeddings or token embeddings. These embeddings capture the semantic meaning of words and their context in the language.
2. **Transformer Architecture:** The core architecture of most modern LLMs is the Transformer. It includes multiple layers of self-attention mechanisms and feedforward neural networks. Transformers enable the model to capture long-range dependencies and relationships between words in a sequence.
3. **Attention Mechanisms:** Within the Transformer architecture, attention mechanisms allow the model to assign different levels of importance (attention weights) to different parts of the input sequence when processing a given token. Self-attention is a fundamental part of understanding context in LLMs.
4. **Positional Encoding:** Since the Transformer doesn't have built-in notions of word order or position, positional encoding is added to the input embeddings to provide information about the position of tokens in the sequence. This helps the model understand word order.
5. **Multi-Head Attention:** LLMs often use multi-head attention mechanisms, where attention is computed in parallel multiple times with different sets of learnable parameters. This allows the model to attend to different aspects of context simultaneously.
6. **Feedforward Neural Networks:** After attention mechanisms, each token's representation is processed through feedforward neural networks, which include activation functions and weight matrices to transform the input.
7. **Layer Normalization and Residual Connections:** These are techniques used to stabilize training in deep neural networks. Layer normalization normalizes activations within each layer, and residual connections help prevent the vanishing gradient problem.
8. **Output Layer:** The final layer of the LLM maps the internal representations back into the vocabulary space, producing a probability distribution over possible output tokens. This allows the model to generate text or make predictions.
9. **Vocabulary and Tokenizer:** LLMs have a predefined vocabulary of words or subwords. A tokenizer is used to break text into tokens that are part of the vocabulary. Out-of-vocabulary words may be handled using subword tokenization.
10. **Pre-training and Fine-tuning:** LLMs are pre-trained on massive text corpora to learn language patterns and knowledge. After pre-training, they can be fine-tuned on specific tasks, such as text classification, translation, or question-answering.
11. **Attention Masking:** In certain applications, such as text generation, an attention mask may be used to prevent the model from attending to future tokens or certain positions in the input.

# Brief introduction to Large Language Models (LLMs)

## Main uses (and emergent properties)

- information retrieval
  - e.g., "What is the capital of France?"
    - The capital of France is Paris.
- text generation/translation
  - e.g., "Write a one sentence definition of corpus linguistics."
    - Corpus linguistics is the study of language using large collections of written or spoken texts to analyze and understand linguistic patterns and phenomena.
  - e.g., "Translate the above sentence into Japanese."
    - コーパス言語学は、大量の書かれたまたは話されたテキストのコレクションを使用して言語のパターンや現象を分析し理解するための言語学の研究です。
- Chatbots (Conversational AI)
  - e.g., "How can I help you?"
- Code generation
  - e.g., "Write a python script to generate a word list from a folder of text files.'

OpenAI Meta AI Google AI

# Brief introduction to Large Language Models (LLMs)

## Main uses (and emergent properties)

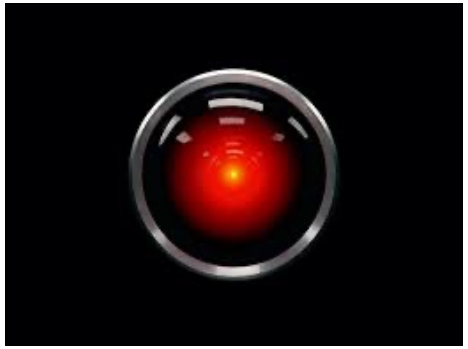
- information retrieval
  - e.g., "What is the capital of France?"
    - The capital of France is Paris.
- text generation/translation
  - e.g., "Write a one sentence definition of corpus linguistics."
    - Corpus linguistics is the study of language using large collecti
      to analyze and understand linguistic patterns and phenomena
  - e.g., "Translate the above sentence into Japanese."
    - コーパス言語学は、大量の書かれたまたは話されたテキスト
      のパターンや現象を分析し理解するための言語学の研究で
- Chatbots (Conversational AI)
  - e.g., "How can I help you?"
- Code generation
  - e.g., "Write a python script to generate a word list from a folder a

python Copy code

```python
import os
import glob
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords

# Ensure NLTK stopwords are downloaded
nltk.download("stopwords")
nltk.download("punkt")

# Define the folder containing text files
folder_path = "your_folder_path_here"  # Replace with your folder path

# Initialize an empty set to store unique words
word_set = set()

# Define a function to tokenize text and add words to the set
def process_text_file(file_path):
    with open(file_path, "r", encoding="utf-8") as file:
        text = file.read()
        tokens = word_tokenize(text)
        # Remove stopwords and non-alphabetic words
        words = [word.lower() for word in tokens if word.isalpha() and word.
        word_set.update(words)

# Iterate through all text files in the folder
for file_path in glob.glob(os.path.join(folder_path, "*.txt")):
    process_text_file(file_path)

# Convert the set to a sorted list
word_list = sorted(list(word_set))

# Print the word list
for word in word_list:
    print(word)
```

# Challenges in using LLMs for corpus-based research

limitations, concerns, and dangers; practical difficulties

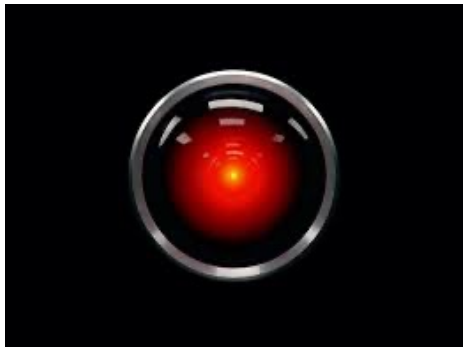


# Challenges in using LLMs for corpus-based research

## Limitations, concerns, and dangers

- 'Hallucinations'
  - i.e., producing an output that is false (or does not match the user's intent)
- Security
  - e.g., leaking private information, helping with phishing scams, …
- Bias
  - e.g., promoting stereotypes, positions of racism, gender bias, …
- Consent
  - e.g., ignoring copyright during training, plagiarizing, stealing personal data, …
- Deployment, maintenance, and scaling
  - e.g., training LLMs requires massive amounts of computing power, time, and energy
- …

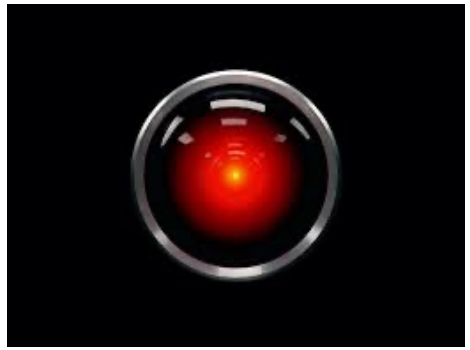
https://www.elastic.co/what-is/large-language-models



# Challenges in using LLMs for corpus-based research

## Limitations, concerns, and dangers - examples of hallucinations

- Q: Who developed the AntConc software tool?
  - AntConc is a tool developed by Future Growth Software, Inc. (StableLM-Tuned-Alpha-7b)
  - AntConc is developed by DIGIn, which stands for David Ingram, an American computer scientist and entrepreneur. (Dolly-v2-12b)
  - AntConc is developed by Ann Taylor, a Professor of Linguistics at the University of British Columbia… (Llama-13b)
  - The developer of AntConc is David Robinson, a professor in the Department of Linguistics at the University of Texas at Austin. (Apaca-13b)
  - AntConc is a corpus analysis tool developed by Mark Davies… (Vicuna-13b)
  - AntConc is a tool for corpus linguistics developed by the University of Liverpool. It was created by a team of linguists led by Professor John Sinclair… (Koala)
  - …

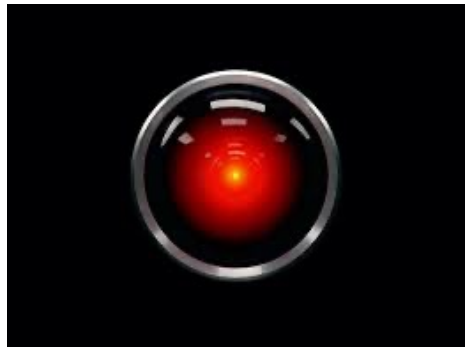


# Challenges in using LLMs for corpus-based research

## Limitations, concerns, and dangers

"reliability is currently the single biggest obstacle for these neural networks being useful... truly useful."

Ilya Sutskever, Chief Scientist of OpenAI, March 2023



# Challenges in using LLMs for corpus-based research

## Practical difficulties

- Knowing  the data
- Authenticity
- Replicability
- Multimodality
- Safety
- Hallucinations
- Active vs passive learning

Crosthwaite, P., & Baisa, V. (2023). Generative AI and the end of corpus-assisted data-driven learning? Not so fast!. *Applied Corpus Linguistics*, *3*(3), 100066.

# Challenges in using LLMs for corpus-based research

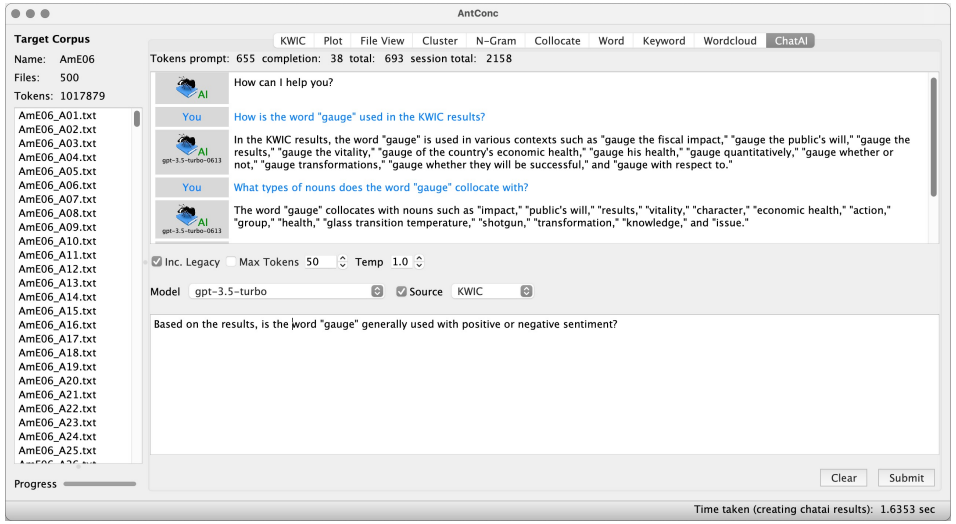
## Practical difficulties

- **Knowing the Data** – LLMs do not give learners or teachers ownership of their data.
- **Authenticity** - LLM data may lack context or register appropriateness (especially crucial for language learners).
- **Replicability** - LLM data does not allow easy replication of findings.
- **Multimodality** - Integrating visualizations with LLM tools can be challenging.
- **Safety** - LLMs raise concerns about data privacy
- **Hallucinations** - LLMs may produce inaccurate or invented terms
- **Active vs. Passive Learning** - LLMs may lead to passive learning, with users potentially copying and pasting output without deep understanding.

Crosthwaite, P., & Baisa, V. (2023). Generative AI and the end of corpus-assisted data-driven learning? Not so fast!. *Applied Corpus Linguistics*, *3*(3), 100066.

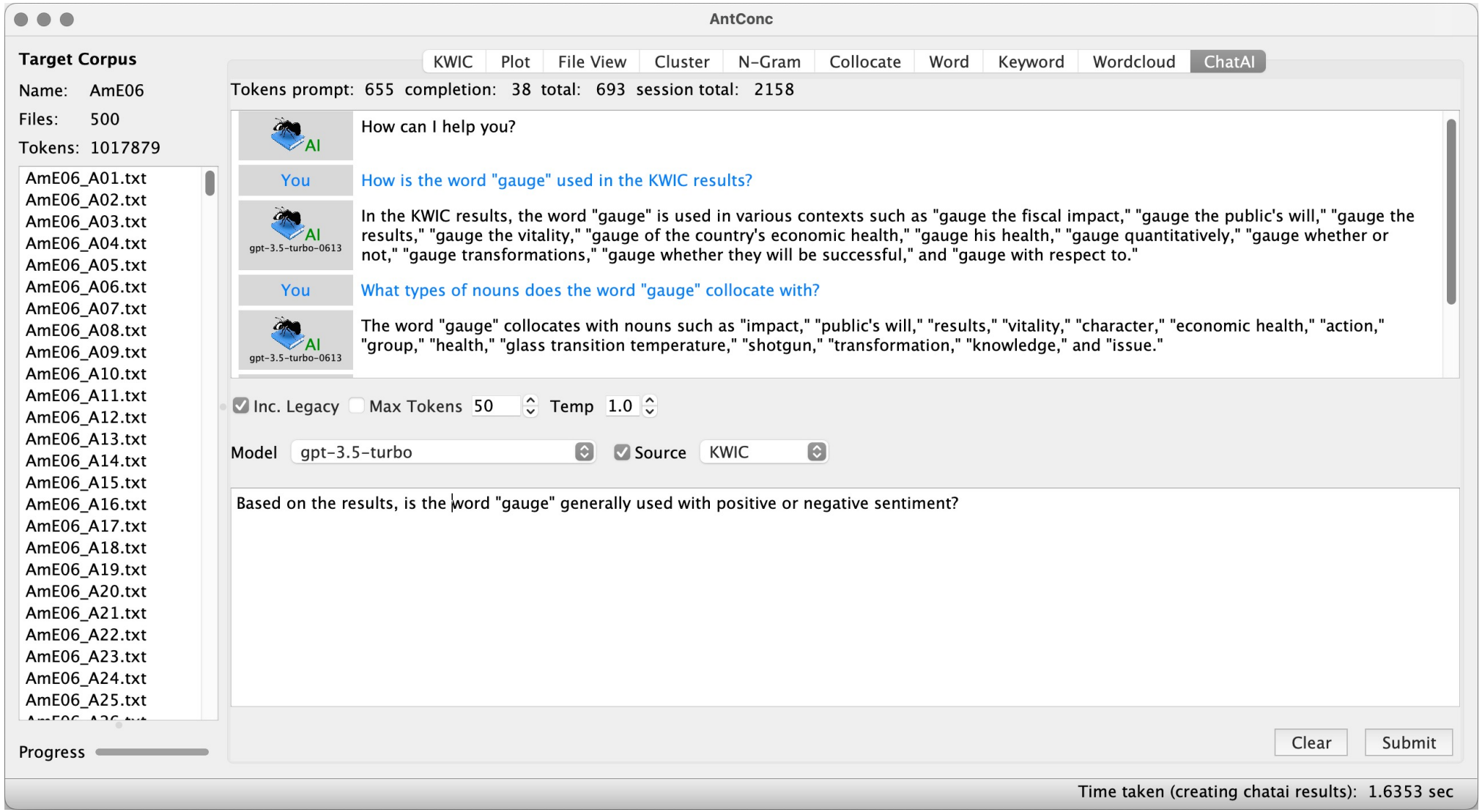
# Integrating LLMs into a corpus analysis toolkit (AntConc)

strategy and design; live demonstration; next steps



# Integrating LLMs into a corpus analysis toolkit (AntConc)

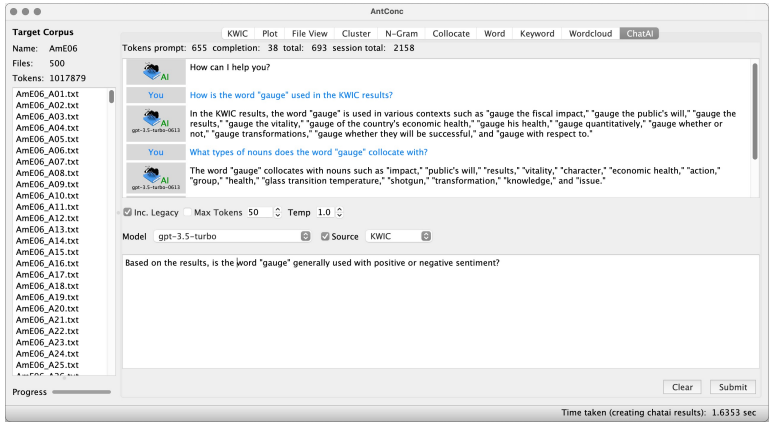
## Strategy and design



# Integrating LLMs into a corpus analysis toolkit (AntConc)

## Strategy and design

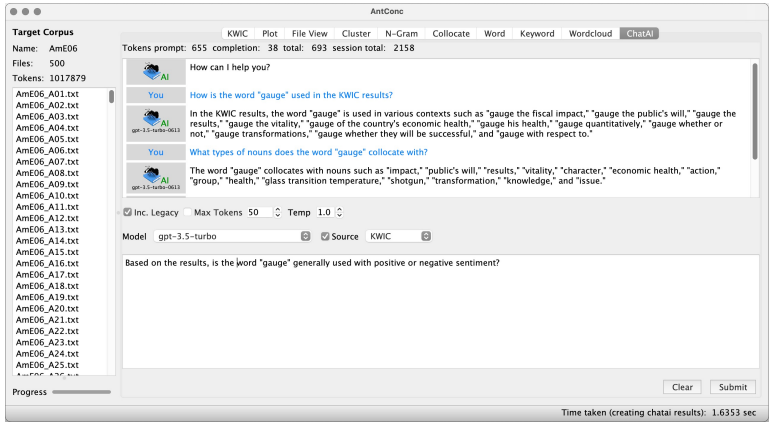
- Easy and direct access to multiple LLMs (commercial and open-source)
  - e.g., GPT-3.5, GPT-4, BARD, LlaMA, Falcon, …
- Simple (user) control of main LLM settings
  - maximum output LLM tokens (e.g., 50)
  - temperature – degree of randomness/creativity in the output
  - system prompt (e.g., "You are an expert corpus linguist")
  - user prompt (e.g., "What in the meaning of 'gauge'?)
- LLM chat using corpus tool results
  - e.g., "What types of nouns does 'gauge' collocate with?" (KWIC)
  - e.g., "What part-of-speech does 'gauge' cluster with?" (CLUSTER)
  - e.g., "What is the general sentiment of the text?" (FILE VIEW)
  - …



# Integrating LLMs into a corpus analysis toolkit (AntConc)

## Strategy and design

- Easy and direct access to multiple LLMs (commercial and open-source)
  - via APIs (for commercial LLMs) and a model repository (for open-source LLMs)
- Simple (user) control of main LLM settings
  - via standard widget controls and menu settings
- LLM chat using corpus tool results
  - via a drop-down list of tool options
    [KWIC, Plot, File View, Cluster, N-Gram, Collocate, Word, Keyword, Wordcloud]
  - via a html-aware text entry widget

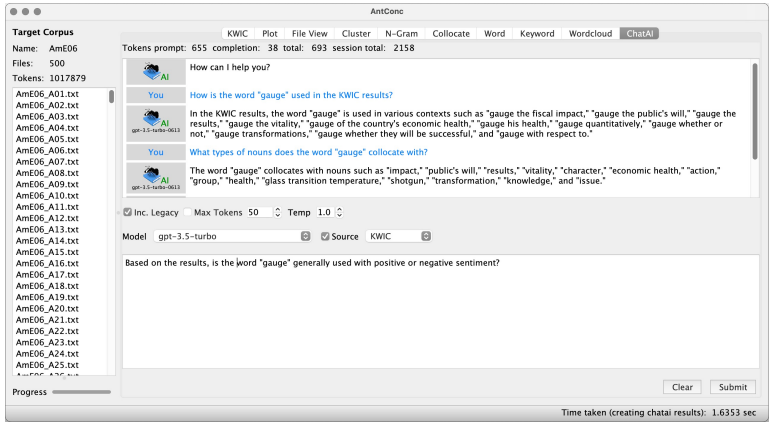


# Live Demonstration

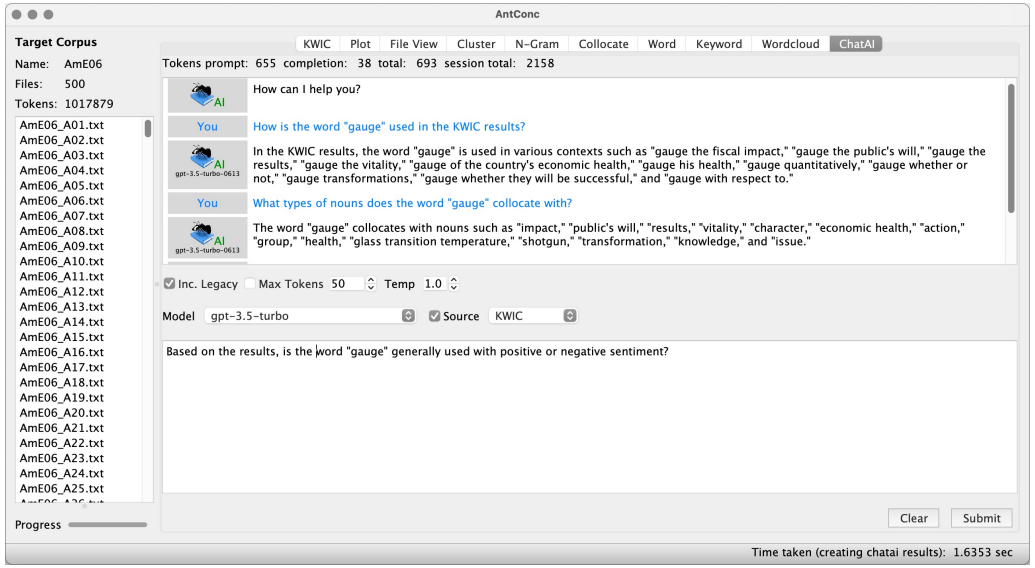
# Integrating LLMs into a corpus analysis toolkit (AntConc)

## Next steps

- Release a development version of AntConc
- Get feedback (bug reports, suggestions, concerns) from users
- Add additional features
  - a 'prompt library' for storing and retrieving useful prompts
  - corpus database updates through natural language
  - corpus creation from LLM output
  - integration of LLM output with corpus tools
  - …



# Summary


AntConc
Target Corpus
Name: AmE06
Files: 500
Tokens: 1017879
AmE06_A01.txt
AmE06_A02.txt
AmE06_A03.txt
AmE06_A04.txt
AmE06_A05.txt
AmE06_A06.txt
AmE06_A07.txt
AmE06_A08.txt
AmE06_A09.txt
AmE06_A10.txt
AmE06_A11.txt
AmE06_A12.txt
AmE06_A13.txt
AmE06_A14.txt
AmE06_A15.txt
AmE06_A16.txt
AmE06_A17.txt
AmE06_A18.txt
AmE06_A19.txt
AmE06_A20.txt
AmE06_A21.txt
AmE06_A22.txt
AmE06_A23.txt
AmE06_A24.txt
AmE06_A25.txt
Progress
KWIC
Plot
File View
Cluster
N-Gram
Collocate
Word
Keyword
Wordcloud
ChatAI
Tokens prompt: 655 completion: 38 total: 693 session total: 2158
AI
How can I help you?
You
How is the word "gauge" used in the KWIC results?
AI
gpt-3.5-turbo-0613
In the KWIC results, the word "gauge" is used in various contexts such as "gauge the fiscal impact," "gauge the public's will," "gauge the results," "gauge the vitality," "gauge of the country's economic health," "gauge his health," "gauge quantitatively," "gauge whether or not," "gauge transformations," "gauge whether they will be successful," and "gauge with respect to."
You
What types of nouns does the word "gauge" collocate with?
AI
gpt-3.5-turbo-0613
The word "gauge" collocates with nouns such as "impact," "public's will," "results," "vitality," "character," "economic health," "action," "group," "health," "glass transition temperature," "shotgun," "transformation," "knowledge," and "issue."
Inc. Legacy
Max Tokens 50
Temp 1.0
Model
gpt-3.5-turbo
Source
KWIC
Based on the results, is the word "gauge" generally used with positive or negative sentiment?
Clear
Submit
Time taken (creating chatai results): 1.6353 sec

# Summary

- Large Language Models (LLMs)….
  - offer interesting and powerful news ways to investigate language data
  - offer many advantages over traditional corpus tools
  - exhibit many limitations, concerns, and dangers
- Integrating LLMs with traditional corpus methods…
  - overcomes many of the existing problems associated with LLMs
  - addresses common problems when using corpora in the classroom
  - introduces exiting new ways to analyze corpora for language research
- AntConc integrates LLMs with traditional corpus methods through its new 'ChatAI' tool

**Laurence Anthony**
anthony@waseda.jp