LLM Text Classification

Lauren Leek
laurenleek.eu

October 24, 2025

Overview and Context

- **Goal of this session:** Introduce text classification with LLMs, recognise strengths and limitations, apply to real-world data.
- **Structure:**
  1. Introduction and theoretical foundation
  2. Hands on tutorial using Google Colab
  3. Advanced topics and best practices
- **Resources:**
  - Colab notebook: `colab.research.google.com`
  - GitHub repository: `github.com/laurencleek/text_classification_workshop`
  - Slides (PDF): available in the Github Repo

Introduction

What is Text Classification?

- **Definition:** The process of automatically assigning predefined categories or labels to text data.
- **Goal:** Transform unstructured text into structured information that can be analyzed quantitatively.
- Grimmer et al.,: humans are great in analysing a straw of hay but humans struggle with organising the haystack.
- **Common applications:**
  - Sentiment analysis (positive/negative/neutral)
  - Topic identification (e.g., monetary vs. fiscal policy)
  - Author or institution profiling
  - Detection of bias, ideology, or stance
- **Why it matters in research:**
  - Enables large-scale analysis of textual corpora (e.g., speeches, policy documents, media).
  - Bridges qualitative and quantitative methods.

Text Classification in Academic Research

- **Political Science:** Classifying legislative speeches or manifestos to study ideology and agenda setting.
- **Economics:** Measuring central bank communication tone, uncertainty, or policy focus.
- **Sociology:** Detecting narratives or framing in social media and news.
- **Linguistics:** Studying variation in discourse across contexts or speakers.
- **Key advantage:** Scales up traditional qualitative analysis through computational pipelines. Supports theory testing.

Example: Central Bank Independence and Communication

- Research Question: *How does central bank independence shape the way central banks communicate?*
- Data:
  - 100+ central banks, 1997-2023.
  - Over 18,000 speeches and statements.
  - Processed with transformer-based language models (Propriety Gemini and GPT-family models).
- Methodology:
  - Classification of speech segments into *Monetary*, *Fiscal*, and *Financial Dominance* to detect policy pressures
  - Causal analysis of agenda shifts following independence event reforms (DiD and IV approach).

Illustrative Pipeline

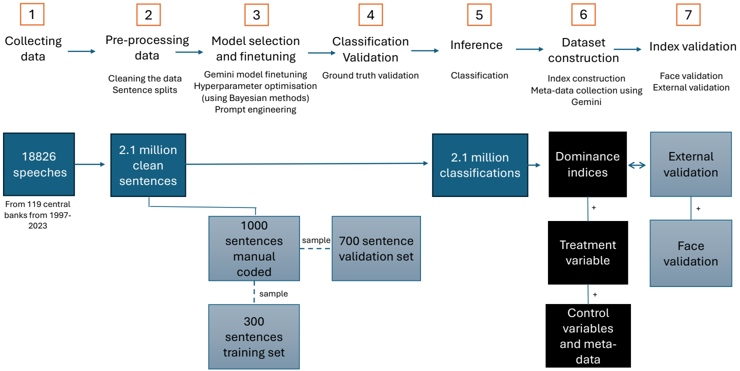


Figure: LLM-based pipeline to classify central bank speeches

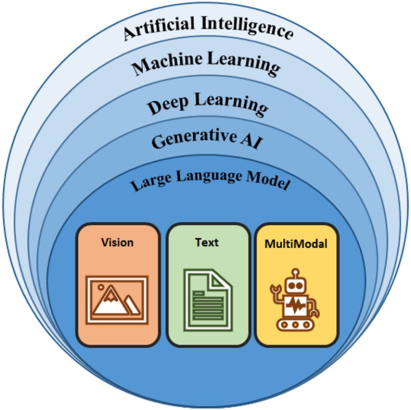

NOOO YOU CAN'T JUST MIX UP ALL THE STEPS OF YOUR TASK AND ASK AN LLM TO DO IT ALL. HOW WILL YOU EVER MAKE A RELIABLE AND EXTENSIBLE SYSTEM THAT WAY?
imgflip.com
HAHA LLM GO BRRR

Take a few minutes to write down your pitch for using LLMs text classification

- What is the **research question** or problem you want to explore?
- What **data or text sources** would you use?
- How could **LLMs help** answer this question or process the data?
- Are there **alternative approaches** or complementary methods?

(We'll discuss a few examples together afterwards.)

Theoretical backbone


Artificial Intelligence
Machine Learning
Deep Learning
Generative AI
Large Language Model
Vision
Text
MultiModal

Introduction to Large Language Models (LLMs)

- **Definition:** Neural networks trained on vast text corpora to predict the next word in context.
- **Architecture:** Based on the Transformer (Vaswani et al., 2017) - relies on *self-attention* to model relationships between all words simultaneously.
- **Capabilities:**
  - Learn general linguistic and world knowledge.
  - Generate coherent text, summarize, or classify.
  - Handle multiple tasks through prompting rather than retraining.
- **Paradigm shift:** From *training separate models* for each task $\rightarrow$ *using one general model* adapted via natural-language instructions.

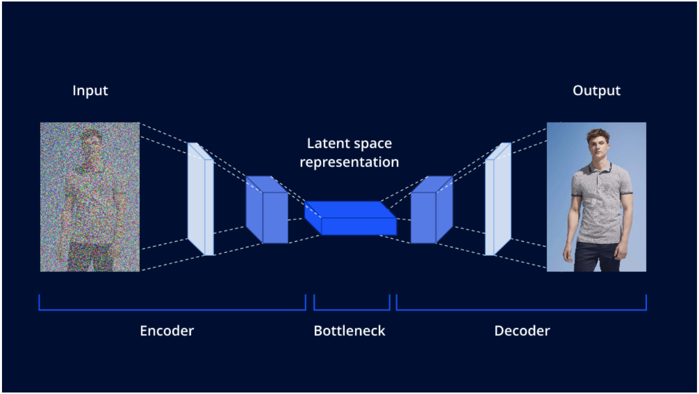
Tokenization: How LLMs Read & Write

- **What is a token?** A minimal unit of text used by the model. Not the same as a word:
  - Can be a whole word (*rare*), punctuation, whitespace, or even part of a word.
  - Models never see raw characters or words; they see **token IDs** (integers).
- **Pipeline:** Text $\to$ *tokenizer* $\to$ token IDs $\to$ model
  Model outputs token IDs $\to$ *detokenizer* $\to$ text.
- **Why it matters:**
  - *Context length* limits are in tokens (not words).
  - *Latency & cost* scale with token count.
  - Prompt wording affects tokenization efficiency (e.g., extra spaces/punctuation add tokens).
  - Some commonly failure modes of LLM linked to tokenisation (e.g., counting or swapping characters)
- **Rule of thumb:** In English, $\sim$4 characters $\approx$ 1 token (but varies by language/script).


GENERATIVE AI
"Let's see who you really are!"
TRANSFORMER

The Transformer Model Underlying LLMs

- **Core idea:** The Transformer (Vaswani et al., 2017) replaces recurrence with *self-attention*, allowing the model to process all words in a sequence simultaneously.
- **Intuition:** Instead of reading word by word, transformers look at the entire sentence at once and decide which words matter most for predicting the next. Click here for a graphical representation.
- **Key components:**
  - **Encoder:** Reads the input text and generates contextual embeddings.
  - **Decoder:** Produces output tokens one by one, attending to both prior outputs and encoder representations.
  - **Self-Attention:** Each word attends to all others, weighting their relevance dynamically.
- **Advantages:**
  - Captures long-range dependencies and contextual nuance.
  - Highly parallelizable - enables large-scale training.
  - Scales effectively to billions of parameters (LLMs).


Input
Output
Latent space
representation
Encoder
Bottleneck
Decoder

Understanding Model Architectures

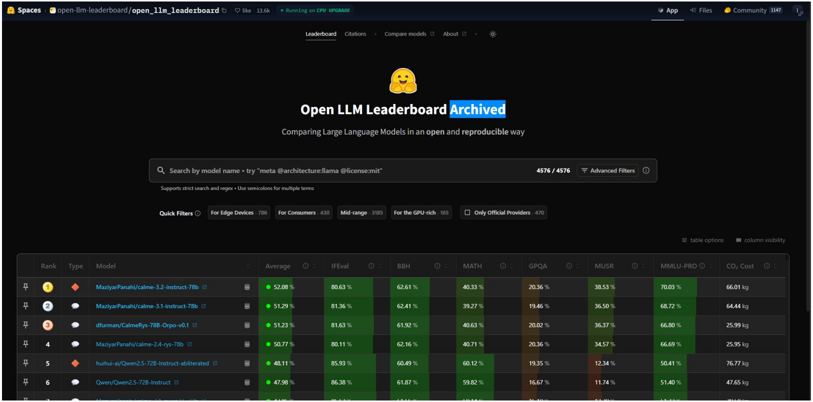
- **Encoder Models** (e.g., BERT, RoBERTa)
  - Encode input text into dense representations.
  - Good for classification, clustering, semantic search.
  - Architecture: bidirectional attention captures context from both directions.
- **Decoder Models** (e.g., GPT series)
  - Predict next tokens autoregressively.
  - Best suited for text generation and completion.
  - Capture strong world knowledge through large-scale pretraining.
- **Encoder - Decoder Models** (e.g., T5, BART)
  - Encode input into a latent space, then decode into output sequence.
  - Useful for summarization, translation, or generation with conditioning.

How LLMs Enhance Text Classification

- **Zero-shot and few-shot learning:** Classify text using plain-language prompts, without labeled training data.
- **Contextual understanding:** Capture meaning, tone, and stance across long passages.
- **Flexible outputs:** Can return labels, explanations, or structured data (e.g., JSON).
- **Adaptability:** Easily extended to new domains like academic text, policy documents, or open responses.

Basic Concepts of Text Classification

- **Goal:** Assign predefined categories (labels) to text documents.
- **Types of classification:**
  - Binary (e.g., positive vs. negative sentiment)
  - Multi-class (e.g., policy areas or topics)
  - Multi-label (texts can belong to several classes)
- **Core idea:** Convert unstructured text into numerical representations (features) suitable for machine learning.
- **Typical pipeline:**
  1. Preprocess text (cleaning)
  2. Vectorize (bag-of-words, TF–IDF, embeddings)
  3. Train or apply a classifier
  4. Validate, validate and validate


Spaces
open-llm-leaderboard/open_llm_leaderboard
like 13.6k
Running on CPU UPGRADE
App
Files
Community 1147
Leaderboard
Citations
Compare models
About
Open LLM Leaderboard Archived
Comparing Large Language Models in an open and reproducible way
Search by model name • try "meta @architecture:llama @license:mit"
4576 / 4576
Advanced Filters
Supports strict search and regex • Use semicolons for multiple terms
Quick Filters
For Edge Devices 786
For Consumers 430
Mid-range 3185
For the GPU-rich 165
Only Official Providers 470
table options
column visibility
Rank Type Model Average IFEval BBH MATH GPQA MUSR MMLU-PRO CO₂ Cost
1 MaziyarPanahi/calme-3.2-instruct-78b 52.08 % 80.63 % 62.61 % 40.33 % 20.36 % 38.53 % 70.03 % 66.01 kg
2 MaziyarPanahi/calme-3.1-instruct-78b 51.29 % 81.36 % 62.41 % 39.27 % 19.46 % 36.50 % 68.72 % 64.44 kg
3 dfurman/CalmeRys-78B-Orpo-v0.1 51.23 % 81.63 % 61.92 % 40.63 % 20.02 % 36.37 % 66.80 % 25.99 kg
4 MaziyarPanahi/calme-2.4-rys-78b 50.77 % 80.11 % 62.16 % 40.71 % 20.36 % 34.57 % 66.69 % 25.95 kg
5 huihui-ai/Qwen2.5-72B-Instruct-abliterated 48.11 % 85.93 % 60.49 % 60.12 % 19.35 % 12.34 % 50.41 % 76.77 kg
6 Qwen/Qwen2.5-72B-Instruct 47.98 % 86.38 % 61.87 % 59.82 % 16.67 % 11.74 % 51.40 % 47.65 kg

Hands-on tutorial

Roadmap

- **Goal:** Build a minimal pipeline for classifying sentences as *descriptive* vs *normative* using an LLM (via the OpenAI API).
- **Artifacts:**
  - Colab notebook (with install cell & runnable pipeline)
  - Validation sample (`validation_sample_limited.xlsx`)
  - Functions for classification via OpenAI API
  - Quick validation (accuracy on a small subset)
- **Links:**
  - GitHub: `github.com/laurencleek/text_classification_workshop`
  - Open in Colab badge at top of notebook

Classification Function (OpenAI)

- **Function:** `classify_sentence(sentence, model='gpt-5-nano')`
- **Prompting:** System role defines task; user message asks to classify a sentence as *descriptive* or *normative*.
- **Parsing:** Extract model reply, normalize to lowercase, map to two labels.
- **Robustness:** Returns `None` if reply not clearly mappable; try/except for API errors.

Demo Run & Results

- **Subset inference:** Apply to first 5 sentences and store in `predicted_label`.
- **Observed example:** All 5 matched ground truth (`accuracy and F1)`.
- **Next step:** Extend to full dataset (rate limits & cost awareness).


Bernie
I am once again asking
for a validation set
imgflip.com

Validation

- **Basic validation:** Compare `label` vs `predicted_label`; compute accuracy/F1.
- **Accuracy:** Measures the proportion of correct predictions.

$$= \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{TP + TN}{TP + TN + FP + FN}$$

  - Works well when classes are balanced.
- **F1 Score:** Harmonic mean of **Precision** and **Recall**.

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

  - **Precision** $= \frac{TP}{TP+FP}$ (How many predicted positives are correct?)
  - **Recall** $= \frac{TP}{TP+FN}$ (How many actual positives were found?)
  - Balances false positives and false negatives - useful for imbalanced datasets.

Ethics and Best practices

Best Practices for Preparing Text Data

- **Data cleaning:** Remove irrelevant symbols, normalize case, handle missing or duplicate entries.
- **Tokenization:** Split text into words, subwords, or sentences (depends on model type).
- **Stopword handling:** Consider whether removing them improves or harms classification accuracy.
- **Balanced classes:** Avoid strong label imbalance - consider reweighting or sampling.
- **Metadata integration:** Add contextual info (e.g., author, date, institution) if relevant.
- **Reproducibility:** Document preprocessing steps and maintain consistent random seeds.

Common Pitfalls and Limitations

- **Data leakage:** Overlap between training and test sets inflates accuracy.
- **Interpretability:** Hard to trace why a model assigns a particular label.
- **Bias and fairness:** Pre-trained models reflect cultural and institutional biases.
- **Prompt sensitivity:** LLM outputs depend strongly on phrasing and context.
- **Evaluation caution:** Use multiple metrics (accuracy, F1, precision-recall) with human or external validation.

LLMs: The Good, the Bad, and the Ugly

Good

- Often accurate on first attempt (*zero-shot learning*).
- Useful for classification without labelled data.
- Can be improved by prompt engineering or fine-tuning.

Bad

- Not deterministic - same input can yield different outputs.
- Sometimes factually wrong yet plausible.
- Energy-intensive: GenAI $\approx$0.5% of global energy use.

Ugly

- Produce *hallucinations*: confident but false outputs.
- "Bullshit" (Frankfurt, 2005): indifferent to truth or falsity.
- Example: invented but credible references or facts.
- Always verify model outputs empirically.

Using LLMs Responsibly in Research

- Evaluate model performance against:
  - Human-coded benchmarks (inter-annotator agreement)
  - Traditional automated classifiers
  - A baseline of no classification
- Treat outputs as **probabilistic**, not factual.
- Always document:
  - Model version, date, and parameters.
  - Verification steps and error rates.
- If affordable, bootstrap, i.e., run the model multiple times

Warnings and Ethical Considerations

- **Data Bias:** Pretrained models may reflect political, cultural, or linguistic biases.
- **Transparency:** Black-box nature of transformer models can obscure causal mechanisms.
- **Reproducibility:** API-based models (e.g., GPT) can evolve over time (sometimes hidden); top-k and temperature settings help somewhat. Still stochastic though!
- **Ethics of Communication Analysis:**
  - Respect institutional confidentiality and policy sensitivity.
  - Avoid normative judgments based solely on model inferences.
  - E.g., Reddit r/ChangeMyView paper retraction

Advanced Topics

Choosing a Specific Model

- *Task Requirements:* Is it a well-defined task with ample labeled data (Task-Specific or Supervised Embedding-Based)? Or a zero-shot scenario (Zero-shot Embedding-Based or Generative with prompting)?
- *Data Availability:* Amount of labelled data influences choice between fine-tuning (Task-Specific) and using pre-trained embeddings or prompting (Embedding-Based, Generative).
- *Computational Resources:* Model size and complexity vary; larger models (some Generative) require more resources.
- *Performance Needs*: Evaluate different models on your specific data (as shown in the notebook) to see which performs best.
- *Interpretability*: Some models (e.g., Logistic Regression on embeddings) might be more interpretable than complex end-to-end models.
- *Cost API Usage*: Using models via APIs can incur costs.

Improving Classification Accuracy: Fine-tuning

- **Concept:** Adapt a pretrained model to a specific domain or task using additional labeled data.
- **Why it helps:**
  - Leverages general linguistic knowledge from large corpora.
  - Aligns representations with domain-specific vocabulary and semantics.
- **Typical workflow:**
  1. Select a suitable pretrained base model (e.g., BERT, RoBERTa, T5).
  2. Add a classification head (e.g., linear or softmax layer).
  3. Train on labeled examples with a smaller learning rate.
  4. Monitor validation performance and apply early stopping.


ALL LLM
FINE-TUNING
Low Rank Adaptation (LoRA)
imgflip.com

Improving Classification Accuracy: Hyperparameter Tuning

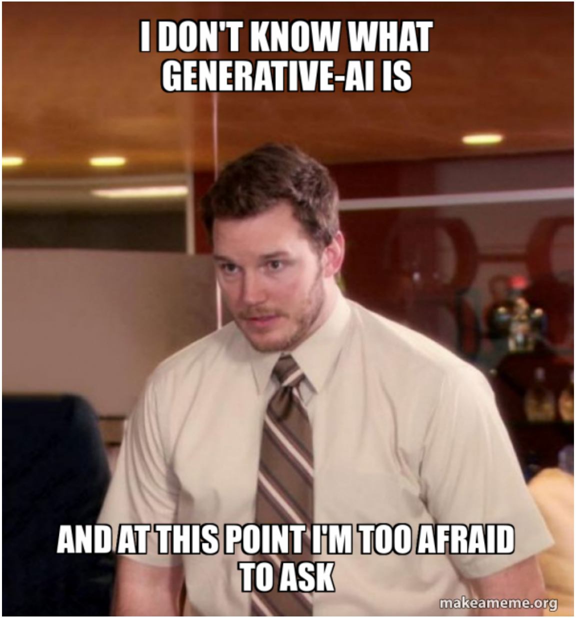
- **Goal:** Optimize training parameters that govern learning efficiency and generalization.
- **Key hyperparameters:**
  - Learning rate, batch size, number of epochs
  - Dropout rate, weight decay, optimizer type
  - Number of unfrozen layers during fine-tuning
- **Search strategies:**
  - Grid search (systematic combinations)
  - Random search (efficient for large spaces)
  - Bayesian optimization (probabilistic, adaptive)

Improving Classification Accuracy: Best Practices

- **Evaluation metrics:**
  - Go beyond accuracy - report precision, recall, (macro) F1, AUC.
  - Use confusion matrices to identify misclassification patterns.
- **Experiment tracking:** Tools like `Weights & Biases`, `MLflow`, or `TensorBoard` for logging results.
- **Iterative process:** PEft fine-tuning and hyperparameter tuning iteratively for best performance.

Takeaways

- Transformer-based architectures (Encoder/Decoder/Encoder-Decoder) underpin modern NLP tasks.
- LLMs enable new frontiers in analyzing policy documents, communication, social media data, etc.
- Text classification is only one of the options, but very accessible for researchers!
- Ongoing challenge: balancing performance, interpretability, and ethics in LLM-based research.


I DON'T KNOW WHAT GENERATIVE-AI IS
AND AT THIS POINT I'M TOO AFRAID TO ASK
makeameme.org

Questions?

Thank you for your attention!

l.c.leek@lse.ac.uk