Literature Review on text classification using LLMs

Using LLMs and RAGs for text classification

Norway, Poland, Austria

Introduction

What are LLMs?

Large Language Models (LLMs) are advanced artificial intelligence systems designed to understand and generate human-like text. They are trained on enormous datasets of text, which enable them to perform a wide variety of language tasks such as translation, question answering, and text generatio. Due to their ability to learn language patterns from large amounts of data, LLMs can provide results that are often close to human responses.

The transformative era of Large Language Models began with the Transformers, whose revolutionary self-attention mechanism redefined the efficiency and scalability of deep learning architectures, paving the way for unprecedented advancements in natural language understanding and generation. Transformers are widely used for LLMs because they rely entirely on self-attention mechanisms, allowing them to process words in parallel rather than sequentially. This makes them significantly more efficient than traditional models like recurrent neural networks (RNNs), which process text one step at a time. Additionally, transformers excel at capturing long-range dependencies in text, meaning they can understand context over extended passages better than previous architectures. Their scalability and ability to leverage massive datasets have made them the foundation for modern LLMs, enabling breakthroughs in translation, text generation, and question answering (Vaswani et al., 2017).

Encoder-decoder architecture

The transformer architecture, introduced in the article *Attention Is All You Need* (Vaswani et al., 2017), revolutionized deep learning for natural language processing by replacing traditional sequential models with self-attention mechanisms. At its core, the encoder processes input text by transforming it into rich, contextual representations. Using multiple layers of self-attention, it effectively captures relationships between words, regardless of their distance in a sentence. Meanwhile, the decoder generates output text while attending to both the previously generated words and the encoder's processed information. The encoder-decoder attention allows the model to focus on relevant sections of the input, making text generation more coherent and context-aware. By enabling parallel computation and enhancing the ability to learn long-range dependencies, transformers have surpassed recurrent models, setting the foundation for modern Large Language Models.

Foundation Models

Building on the foundational encoder-decoder structure, modern Large Language Models undergo an extensive pre-training phase. The main goal of the pre-training process for LLMs is to build general linguistic knowledge. A vast amount of unlabeled text data is used in the process of data collection and training on large dataset by predicting next word. The process can take even months, because the entire model must be trained from scratch. The pre-trained models can then be fine-tuned for specialized tasks.

Adapting LLMs to specific tasks

The evolution of machine learning techniques has led to diverse approaches for optimizing model performance and adaptability. Among these, in-context learning, Retrieval-Augmented Generation (RAG), fine-tuning, and transfer learning represent distinct strategies for enhancing the capabilities of LLMs. Understanding the differences and advantages of these techniques is crucial for selecting the most effective approach for a given task.

The main idea of in-context learning (ICL) is to learn from analogy. ICL requires a few examples to form a prompt context. It joins a query question and the piece of prompt context together to form the input. This input is given to the language model for prediction. Different from supervised learning, which requires a training stage that uses backward gradients to update model parameters, ICL does not perform parameter updates. The model is expected to learn the pattern hidden in the given examples and accordingly make the right prediction (Dong et al., 2022).

Another frequently used technique is Retrieval-Augmented Generation (RAG). This is an approach that involves retrieving relevant data and using it as an additional context for the LLM. The model does not depend only on knowledge from the training data, but retrieves relevant information from the provided dataset and uses it as additional context for the LLM. With RAG architecture, organizations can use LLM models and join it with their own database to receive better responses to their prompts. This process is cost and time effective (Parthasarathy et al., 2024). There is no need to fine-tune the model to get the expected answer. Also, in terms of suppressing hallucinations and ensuring accuracy, RAG systems tend to perform better. The probability of generating incorrect information is quite low.

The most complex and challenging techniques are transfer learning and fine-tuning. Transfer learning is a term for ML approaches that leverage knowledge gained from solving one problem (the source domain) to improve speed, efficiency, and data requirements in solving a different but related problem (the target domain). This process involves reusing the already trained parameters of a model, eliminating the need to train a neural network from scratch and allowing it to be adjusted specifically for the new

task. The greatest advantage of this solution is the reduction of costs and resources used to prepare the new ML model (Nadali et al., 2024).

Fine-tuning, in turn, is a specific form of transfer learning that uses a pre-trained model as a foundation. The process involves further training on a smaller, domain-specific dataset. This approach builds upon the model's pre-existing knowledge, enhancing performance on specific tasks with reduced data and computational requirements. Fine-tuning transfers the pre-trained learned patterns of the model and features to new tasks, improving performance and reducing training data needs. As the result the last layers of the model become adapted for a new task – the one that was the goal of the fine-tuning process.

The relations between base, pre-trained and fine-tuned models are presented on the diagram below.

Setups for using LLMs for classification

The most obvious setup for using LLMs for classification is to have an LLM output the category for each item needing to be coded in a statistics production process. However, this can be overly costly when numerous items need to be coded, whether the cost is incurred through purchasing tokens to use a commercial LLM's API, or whether through needing large resources for training or running in-house models. In other cases, it might only be practical to use LLM's with smaller volumes of data, as they might not give reliable enough classifications to use without manual checking. Thus, through presentations at AIML4OS and meetings such as NTTS 2025, we have heard of other ways NSI's have experimented with using LLMs for classification in less direct ways:

- Create or expand the training set by using LLM's to classify existing examples
- Address class imbalance by synthetically creating training examples for rare classes using an LLM
- Use LLM's for feature augmentation to generate additional features for training a model

For example, Hess (2024) shows significant improvements in NACE classification when adding synthetically generated examples to the training set created by GPT-4 Turbo.

Specific issues

Text classification is a common task in statistical offices and is the process of assigning categories to a given text. A HLG-MOS white paper on LLMs for official statistics (2023), mentions specifically classification tasks as a potential field where LLMs can contribute significantly to reduce resource-intense manual tasks.

Using LLMs for text classification has several specific challenges which are briefly explored in this section, including language challenges, prompt engineering, measuring model uncertainty and legal/ethical considerations.

Languages, English/local/multilingual

Large Language Models can be categorized into multilingual and local models. Multilingual models, such as GPT-3 and BERT, are designed to understand and generate text in multiple languages, leveraging vast amounts of data from diverse linguistic sources. This versatility makes them powerful tools for global applications, but they may sometimes lack the cultural and contextual nuances of individual languages. On the other hand, local models, like polish LLM - Bielik, are tailored to specific languages or regions. They are trained on more focused datasets, which allows them to capture the subtleties and intricacies of the local language. This makes them particularly effective for applications requiring deep cultural and contextual understanding. The choice between multilingual and local models depends on the specific needs and goals of a project. Each type of LLM models offers unique advantages.

Lehmann et al. (2020) describe how bilingual transfer learning can be very effective in reusing training data in another language and cultural context. The authors show how to make use of a classifier developed on German data for COICOP classification for the CPI survey for the French CPI survey, using a cross-lingual embedding model. They found a large savings in resources with the transfer learning method over developing a model for a single language classifier from scratch, or over using a classifier with both languages trained jointly.

Prompt engineering

Prompt engineering concerns composing and structuring the instructions to a generative model to produce the best possible output for the task. It is an integral part of the in-context learning process and has gained much attention in the past few years. Most natural language processing (NLP) tasks, including text classification, can be formatted as a question-answering problem. The prompt plays an important part of the model setup and may help to define the output and give context. Buono et al. (2024) provides some general remarks on prompting that it may be useful to provide system instructions to set general conditions on what information should be returned and in what format. Earlier studies have shown that providing a small number of relevant

examples in the prompt (few-shot learners) improves accuracy (Brown, 2020). This may, for example, be in the context of a RAG where the contexts are incorporated into the prompts. Other studies have shown that the order with which the examples are provided can also impact the outcome (Lu et al. al, 2021).

A further development in prompting came in 2022 when Wei et al. (2022) proposed Chain of Thought (CoT) prompting. This involves not only providing some examples, but also a description of the process to solve the problem. This led to other prompting methods including Zero-shot CoT (Kojima, 2022), where only the think process is included and Self-consistency CoT, where the prompt is asked multiple times, and a majority vote is used for the final answer.

Various forms of prompt engineering have been explored in classification tasks. Clavié et al. (2023) explored prompt engineering with LLMs to classify job applications and found zero-shot CoT performed better than few-shot CoT (and other prompting methods) in this setting. Kanvas et al. (2023) also investigated job postings and classified them into ESCO occupation groups using LLMs and found improvements using CoT prompting. Zhang et al. (2024) looked at classification of clinical notes and again found prompts with clear thought steps (CoT) improved the accuracy of the classifications. Specific examples of testing different prompting methods in NSIs is sparce, and is an area for further investigation.

Zhao et al. (2021) describes how unstable few-shot learning can be, with large variations in accuracy caused by changing the prompt, the choice of the training examples included in the prompt, and even the order of the examples. They found substantial performance improvement from recalibrating the model's output probabilities corresponding to various tokens to be neutral to such artifacts.

Confidence scores from models

The use of machine learning models for automated coding at NSI's has relied not only on the predicted codes, but also on the prediction probabilities, returned by models. The prediction probability gives the probability the model attaches to the predicted classification being correct, and is thus a measure of the model's confidence. A measure of confidence is necessary to sort the items into a group having quality predictions that are automatically labelled, vs a group that needs manual labelling.

Many of the most used LLM's today allow the user to access prediction probabilities. Since LLM's are trained to probabilistically generate words in sequence assuming all previous words, each time they generate a word in the output sequence they calculate a probability distribution over all possible words in their vocabulary. In particular, the probability associated with the word/token that is output can be viewed as a log or logit probability. For coding tasks, the predicted code is typically returned as a single token in the output text and thus has a corresponding prediction probability. The log probability

for OpenAI models is accessible via the “logprobs” parameter in an API call. For NSI’s using locally downloaded models offline, here is some sample code for calculating the prediction probabilities from tokens output from a model on Hugging Face:

```
import torch
from transformers import AutoModelForCausalLM, AutoTokenizer

# Load model and tokenizer
model_name = "gpt2"  # Change this to any GPT model
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(model_name)

input_text = "The quick brown fox"

# Tokenize and convert to tensor
input_ids = tokenizer(input_text, return_tensors="pt").input_ids

with torch.no_grad():
    outputs = model(input_ids)

# Get logits (last token's logits)
logits = outputs.logits[:, -1, :]  # Shape: (1, vocab_size)

# Top-k selection to get logits, indices
k = 5
top_k_probs, top_k_indices = torch.topk(logits, k, dim=-1)

# Convert logits to probabilities using softmax
probs = torch.nn.functional.softmax(top_k_probs, dim=-1)

# Decode top-k tokens
top_k_tokens = [tokenizer.decode(idx.item()) for idx in top_k_indices[0]]

# Print results
for i in range(k):
    print(f"Rank {i+1}: Token='{top_k_tokens[i]}', Probability={probs[0, i].item():.4f}")
```

Some very recent papers study the value of these log probabilities, as well as other potential metrics, to measure the quality of predictions. (Farr et al. 2024) compares 5 intuitively appealing evaluation metrics for an LLM for a coding task. The quality of each metric is evaluated as the area under the curve (AUC) when generating a curve that trades off percent error with percent needing manual classification. The metrics evaluated are: 1) LLM’s quantitative self-report (“*on a scale of 0-100, how confident are you?*”), 2) LLM’s qualitative self-report (“*describe your confidence as low, medium, high, or absolute*”), 3) the (log) prediction probability, 4) confidence scores, which are the difference in log probability between the token with the highest and the token with the second-highest log probability, and thus measure the dominance of the chosen token, 5) a *confidence ensemble* metric, which gives confidence across a set of LLM’s by summing the confidence scores across all LLM’s outputting a particular token. Across a number of LLM’s and datasets, the paper found the confidence ensemble metric performed best, likely because unlike the other metrics, it involved taking a vote over multiple LLM’s. The confidence score method was almost as good overall. The other 3 metrics scored significantly lower, including the very widely used strategy of simply using the prediction probability returned. It was also found that qualitative self-reporting

scored slightly better than using prediction probabilities, and quantitative self-reporting the worst. Other works such as (Tian et al. 2023) found the same result that qualitative self-verbalization from an LLM on its confidence outperformed log probabilities.

The papers mentioned above relied on black box methods for uncertainty quantification; that is, methods for quantifying performance with access to only the final response generated by the LLM. In contrast, white box methods allow access to the last activation layer of the LLM which generates the output through sampling. (Becker and Soatto 2024) show a way to calculate confidence scores by sampling over the possible output answers and attached explanations. They use Bayesian formula to calculate a probability attached to answers through the likelihood of the explanations. (Sun et al. 2024) study metrics from both white-box and black-box methods for confidence estimation, using area-under-the-curve measures to evaluate the metrics. They found that a combination can work the best, and that an effective self-probing strategy where the prompt asked "How likely is the below state to be correct? Analyze the state, provide a brief reason, and give confidence (0-1)" improved the calibration level of confidence scores.

There is also an increasingly large literature on using LLM's themselves for evaluation. For example, Potsawee (2023) describe the "SelfCheckGPT" tool, in which confidence is assessed through sampling from a GPT multiple times and verifying that the outputs (classifications) are the same.

Risks associated with using of pre-trained LLM models

Eurostat's recent report on LLMs (Buono, Felecan, & Tessitore, 2024) provides a good overview of LLMs and their use in official statistics. It also outlines some key issues and concerns regarding the use of pre-trained LLMs at NSIs. The topics of bias, transparency and security are all relevant concerns in text classification tasks. Bias in LLMs occurs because of the underlying biases in texts used to train the models. For example, Bas (2024) looked at gender roles in occupations and found biases in pre-trained LLMs when compared to both neutrality and statistical labor market benchmarks.

Transparency of LLMs is limited and they are often regarded as a "black box". The models have become so complex, that there is a real lack of explainability and understanding on how they work. This can be problematic for tasks in official statistics, as transparency is fundamental to the integrity of NSIs (UNECE, 2025).

Finally, security issues, such as those associated from attacks on LLMs, may pose a problem in the future. Both proprietary and open-source models may be susceptible to security issues. For example, model poisoning refers to attacks where malicious data is injected into training data sets and can lead models to learn incorrect outcomes (Wang et al., 2023). Prompt leaking, where private data may be disclosed and prompt injection,

where malicious inputs led to models performing unwanted instructions may also be security issues in deploying LLMs in NSIs (UNECE, 2025).

Use cases

Traditional machine and deep learning approaches have been widely used for standardized code classification. A recent study on text classification with machine learning in labor market intelligence involved the classification of occupations using ISCO codes (Boselli et al., 2024), where they show that classical machine learning models, such as Support Vector Machines and Neural Networks, trained on bag-of-words representations, outperform deep learning-based models for ISCO classification. While deep learning models utilizing Word2Vec embeddings combined with Convolutional Neural Networks have been explored, they generally do not surpass bag-of-words and Support Vector Machine approaches in accuracy.

Recent work has explored the use of pretrained embeddings for NACE classification (Le Pera et al., 2024), demonstrating that the inclusion of hierarchical structures in the training data helps reduce embedding errors and improve classification performance. This research highlights the potential of leveraging pretrained embeddings to enhance industry classification tasks.

ClassifAI is an experimental pipeline developed by the British Office for National Statistics, to classify free-text responses into standardized classification systems using a Retrieval-Augmented Generation approach (Office for National Statistics, 2023). The system embeds a knowledge base of classification descriptions and related activities using the MiniLM transformer model. It then embeds survey responses to create a shortlist of relevant classification codes based on semantic similarity. A general-purpose Large Language Model, such as Gemini-Pro, assesses the shortlisted codes and selects the most appropriate classification. The model can also flag responses as "uncodable" and suggest follow-up questions to improve classification quality. Initial testing on de-identified labor market survey responses classified into the Standard Industrial Classification system demonstrated marginal improvements in classification agreement compared to traditional methods.

PLM-ICD is a framework for automatically assigning International Classification of Diseases codes to clinical notes using pretrained language models (Huang et al., 2022). This study highlights three key challenges in ICD classification. The large label space makes multi-label classification complex. Long input sequences in clinical notes often exceed standard pretrained language model input lengths, leading to truncation issues. There is also a domain mismatch, as general domain pretrained language models do not effectively capture medical language nuances. To address these challenges, the study

proposes domain-specific pretraining by fine-tuning models such as BioBERT and PubMedBERT on biomedical texts to improve performance. Segment pooling is employed, where long clinical notes are divided into smaller segments, which are processed separately and aggregated to preserve context. A label-aware attention mechanism is introduced, which learns which text regions are important for each ICD code, enhancing classification accuracy. Experiments on the MIMIC-3 and MIMIC-2 datasets demonstrated that PLM-ICD achieves state-of-the-art performance in automatic ICD coding.

A study on automated occupation coding with hierarchical features explored fine-tuning pretrained models such as BERT and GPT-3 for classification in the German KldB 2010 system (Safikhani et al., 2023*)*. The findings showed that fine-tuned models demonstrated improved accuracy in classifying job occupations. The study emphasized the importance of high-quality labeled data for enhancing Large Language Model performance. BERT-based models outperformed GPT-3 by approximately 34 percent, highlighting the effectiveness of fine-tuning domain-specific transformers.

While traditional machine learning approaches remain competitive in specific classification tasks, Large Language Models provide enhanced flexibility and performance, particularly when paired with retrieval-augmented techniques or hierarchical learning strategies. Future research should continue exploring domain-specific pretraining, hierarchical classification, and attention-based mechanisms to further improve classification accuracy in standardized coding systems.

Conclusions

Large Language Models (LLMs) are advanced AI systems trained on datasets to understand and generate human-like text. LLMs can automate text classification tasks through embeddings and inference models, making them particularly interesting for National Statistical Institutes (NSIs). Pretraining builds general linguistic knowledge, which can be refined through fine-tuning for specific tasks, while Retrieval-Augmented Generation (RAG) architecture can help reduce hallucinations. Several NSI have started testing these models in the contexts of standard classifications such as NACE and ISCO. While traditional methods remain competitive, LLMs offer enhanced flexibility and potential performance benefits, especially with retrieval-augmented, fine-tuning and hierarchical learning strategies. Despite their potential, LLMs face challenges such as language diversity, confidence score quality, and prompt engineering. Techniques like bilingual transfer learning and Chain of Thought prompting may enhance accuracy. There are also risks such as bias, transparency, and security, which must be addressed when deploying LLMs in official statistics.

References

Bas, T. (2024). Assessing Gender Bias in LLMs: Comparing LLM Outputs with Human Perceptions and Official Statistics. *arXiv preprint arXiv:2411.13738*.

Becker, E., Soatto, S. (2024). Cycles of Thought: Measuring LLM Confidence through Stable Explanations. https://arxiv.org/abs/2406.03441

Boselli, R., Cesarini, M., Mercorio, F., & Mezzanzanica, M. (2017). Using machine learning for labour market intelligence. In Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part III 10 (pp. 330-342). Springer International Publishing.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, *33*, 1877-1901

Buono,D., Felecan, M. & Tessitore, C. (2024). *An introduction to Large Language Models and their relevance for statistical offices*. Luxembourg: Publications Office of the European Union.

Clavié, B., Ciceu, A., Naylor, F., Soulié, G., & Brightwell, T. (2023, June). Large language models in the workplace: A case study on prompt engineering for job type classification. In *International Conference on Applications of Natural Language to Information Systems* (pp. 3-17). Cham: Springer Nature Switzerland.

Dong, Q., Li, L., Dai, D., Zheng, C., Ma, J., Li, R., Xia, H., Xu, J., Wu, Z., Liu, T., Chang, B., Sun, X., Li, L., & Sui, Z. (2022). A Survey on In-context Learning. http://arxiv.org/abs/2301.00234

Farr, D., Cruickshank, I., Manzonelli, N., Clark, N., Starbird, K., West, J. (2024). LLM Confidence Evaluation Measures in Zero-Shot CSS Classification. http://arxiv.org/abs/2410.13047

Hess, G. (2024). Use of a large language model to derive the economic sector of businesses from unstructured text on economic activities. Conference on Foundations and Advances of Machine Learning in Official Statistics, Wiesbaden, Germany, 4th April 2024.

Huang, C. W., Tsai, S. C., & Chen, Y. N. (2022). *PLM-ICD: Automatic ICD coding with pretrained language models. arXiv preprint arXiv:2207.05289*.

Kavas, H., Serra-Vidal, M., & Wanner, L. (2024). Enhancing Job Posting Classification with Multilingual Embeddings and Large Language Models

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. (2022). Large language models are zero-shot reasoners. *Advances in neural information processing systems*, *35*, 22199-22213.

Lehmann, E., Simonyi, A., Henkel, L., Franke, J. (2020). Bilingual Transfer Learning for Online Product Classification. In Proceedings of Workshop on Natural Language Processing in E-Commerce, pages 21–31, Barcelona, Spain. Association for Computational Linguistics.

Le Pera, G., Jean, N., & Vidali, A. (2024). Unlocking NACE Classification Embeddings with OpenAI for Enhanced Analysis and Processing.

Lu, Y., Bartolo, M., Moore, A., Riedel, S., & Stenetorp, P. (2021). Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*

Nadali, A., Zhong, B., Trivedi, A., & Zamani, M. (2024). *Transfer Learning for Control Systems via Neural Simulation Relations*. http://arxiv.org/abs/2412.01783

Nguyen, Q. H. (2024). *Automatic evaluation of companies' alignment with EU Taxonomy using Large Language Models* (Master's thesis, University of Twente)

Office for National Statistics. (2023). *ClassifAI: Exploring the use of large language models (LLMs) to assign free text to commonly used classifications*. Data Science Campus. https://datasciencecampus.ons.gov.uk/classifai-exploring-the-use-of-large-language-models-llms-to-assign-free-text-to-commonly-used-classifications/

Parthasarathy, V. B., Zafar, A., Khan, A., & Shahid, A. (2024). The Ultimate Guide to Fine-Tuning LLMs from Basics to Breakthroughs: An Exhaustive Review of Technologies, Research, Best Practices, Applied Research Challenges and Opportunities. http://arxiv.org/abs/2408.13296

Potsawee, M., Liusie, A., Gales, M. (2023) SELFCHECKGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models. 11 Oct 2023. https://arxiv.org/pdf/2303.08896

Safikhani, P., Avetisyan, H., Föste-Eggers, D., & Broneske, D. (2023). Automated occupation coding with hierarchical features: A data-centric approach to classification with pre-trained language models. Discover Artificial Intelligence, 3(1), 6.

Sun, Y.J., Dey, S., Hakkani-Tur, D., Tur, G. (2024) Confidence estimation for llm-based dialogue state tracking. IEEE Spoken Language Technology Workshop (SLT).

Tian, K., Mitchell, E., Zhou, A., Sharma, A., Rafailov, R., Yao, H., Finn, C., Manning, C.D. (2023). “Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback,” In Proceedings of the 2023

Conference on Empirical Methods in Natural Language Processing, pages 5433–5442, Singapore. Association for Computational Linguistics.

UNECE (2025) *Generative AI for Official Statistics* (upubl.)

UNECE (2023) *Large Language Models for Official Statistics*. HLG-MOS white paper. December 2023.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. http://arxiv.org/abs/1706.03762

Wang, J., Liu, Z., Park, K. H., Jiang, Z., Zheng, Z., Wu, Z., ... & Xiao, C. (2023). Adversarial demonstration attacks on large language models. *arXiv preprint arXiv:2305.14950*

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q.V. & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, *35*, 24824-24837

Zhang, X., Talukdar, N., Vemulapalli, S., Ahn, S., Wang, J., Meng, H., ... & Chen, B. (2024). Comparison of Prompt Engineering and Fine-Tuning Strategies in Large Language Models in the Classification of Clinical Notes. *AMIA Summits on Translational Science Proceedings*, *2024*, 478

Zhao, T.Z, Wallace, E., Feng, S., Klein, D., Singh, S. (2021). Calibrate before use: improving few-shot performance of language models. International conference on machine learning, PMLR, 2021, pp. 12697–12706.