Cooking Hallucinations: Dynamic Train-Time Softmax Tempering for PolyCompQA (Polymer Composite QA)

Anonymous ACL submission

Abstract

Large language models (LLMs) can produce overconfident and factually unsupported answers, limiting their reliability for tasks that demand faithfulness to provided evidence. Softmax tempering, which is multiplying pre-softmax logits by a temperature T at training time, was originally used for knowledge distillation, then for offering a simple approach to improve both confidence calibration and factual consistency. In this paper, we provide (1) a structured literature review of softmax tempering in Transformer-based models; (2) an empirical study using Softmax., comparing tempered fine-tuning against standard fine-tuning on SQuAD v2 and a new dataset, PolyCompQA, which contains QA pairs based on polymer composite literature tables. Our experiments reveal that moderate temperatures (e.g., $T = 1.67$) reduce hallucinations and improve calibration metrics, with minimal implementation overhead.[1]

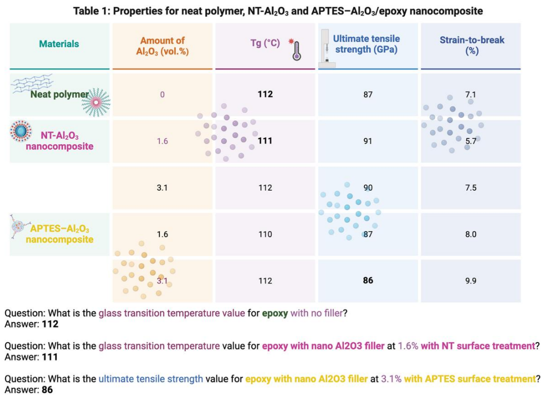

Figure 1: An example of the **PolyCompQA** task, demonstrating the need for materials knowledge and the ability to interpret various components of a table and their relationships in order to answer the questions accurately. For example, "epoxy with no filler" refers to the neat polymer, and "glass transition temperature" corresponds to Tg. Identifying the correct sample requires utilizing information from the first two columns.

1 Introduction

Large Language Models (LLMs) have shown remarkable performance in various NLP tasks, including summarization and question answering (QA) (Brown et al., 2020; Chowdhery et al., 2022), but they often exhibit *hallucination*—overconfidence in incorrect or unsupported answers (Lin et al., 2022; Min et al., 2020; Xu et al., 2024). This issue is particularly problematic in domains where *faithfulness* - adhering closely to textual evidence - is crucial, such as scientific information extraction and QA.

Overconfident but incorrect outputs limit the real-world applicability of LLMs, particularly in settings demanding factual accuracy and interpretability. While methods like retrieval-augmented generation (RAG; Lewis et al., 2020) and self-consistency decoding (Wang et al., 2022) can help mitigate but not entirely remove hallucination, they introduce substantial complexity in implementation or inference.

In this paper, we revisit *train-time softmax tempering*, which modifies the model's training loss to reduce overconfidence by dividing logits by a temperature $T > 1$ (Hinton et al., 2015; Müller and Müller, 2022). In contrast to existing work, we explore softmax tempering during fine-tuning so as to improve performance on downstream tasks. In particular, we investigate the effects of softmax tempering when fine-tuning on two QA datasets: (1) a remix of that we make of SQuAD v2 (Rajpurkar et al., 2018) and (2) a newly-introduced Polymer Composite QA Dataset (PolyCompQA) on scientific paper tables.

As in prior literature, we observe that softmax tempering has dual-edged effects; that is, it can be beneficial in some scenarios while introducing trade-offs in others. We explore this dual behavior

[1]Code and data will be publicly released.

by adapting **Birdie**, a reinforcement learning (RL) procedure that dynamically adjusts hyperparameters during training to optimize the overall the loss reduction to instead maximize accuracy during text generation (Blouir et al., 2024).

Our experiments show that the synergy between softmax tempering and Birdie's adaptive scheduling further reduces hallucinations on SQuAD v2, as well as on PolyCompQA, a new "Research Paper Table QA" dataset of $4,270$ unique question–context pairs. The large-scale analysis we report here on this new dataset demonstrates that temperature tuning strongly influences whether the model can correctly identify missing context versus employing relevant evidence.

Contributions

Our paper provides the following contributions:

- An introduction of PolyCompQA (Polymer Composite QA), a new dataset with $4,270$ unique question-context pairs that focus on property information extraction from published materials science articles;
- An investigation into softmax tempering during fine-tuning on model performance during downstream tasks;
- An empirical demonstration of reduced hallucination and copying issues in scientific information extraction and QA; and
- An exploration on the potential of maximizing softmax tempering by adapting Birdie, a reward-based pretraining procedure to select softmax temperatures during finetuning

The rest of this paper proceeds as follows. Section 2 places our contributions in context of related literature. The methodology is described in Section 3, and experiments are related in Section 4.

2 Related Work

Prior research has highlighted various methods to address hallucination, including RAG (Lewis et al., 2020), self-consistency (Wang et al., 2022), and reranking (Krishna et al., 2022). Calibration strategies, such as post-hoc temperature scaling (Guo et al., 2017) and confidence penalties (Pereyra et al., 2017), aim to align a model's predicted probabilities with actual correctness rates, ensuring that the model's outputs are more reliable. These post-hoc methods focus on adjusting model confidence after training to better match true outcomes. f An alternative approach to model calibration is *train-time softmax tempering*, which directly influences the model's softmax distribution during training. This method modifies the model's training loss by dividing logits by a temperature $T > 1$, reducing overconfidence before predictions are made (Hinton et al., 2015; Müller and Müller, 2022). The division by a temperature effectively smooths the predicted probabilities, encouraging more uncertainty in the model's output, which can help reduce the risk of hallucination, particularly in QA tasks (Wang et al., 2022).

Generalizing softmax tempering, entropy control at training time can benefit convergence with semi-supervised learning, model calibration, and regularization(Guo et al., 2017). In semi-supervised learning, **?** introduced *minimum entropy regularization* to encourage confident cluster assignments for unlabeled data, while Williams and Zipser (1989) adopted *pseudo-labeling* as a practical way to push unlabeled examples toward low-entropy, one-hot predictions. *Label smoothing* (Dorigatti et al., 2024) or *confidence penalty* (Pereyra et al., 2017) increase entropy during training, discouraging the model from peaking its predictions too sharply..

In QA tasks, the importance of avoiding hallucination is heightened by the requirement of faithful, evidence-based answers. Datasets such as SQuAD v2 (Rajpurkar et al., 2018) have highlighted the need to detect unanswerable queries or missing context, and methods that foster better calibration can help models abstain from guessing when unsure. Retrieval-based QA (Lewis et al., 2020) systems can mitigate hallucination by incorporating references to support an answer. However, these solutions can incur complexity at inference time. By contrast, *train-time* softmax tempering (Dabre and Fujita, 2020; Li et al., 2022; Müller and Müller, 2022) requires no greater infrastructure when deploying a model, as it directly modifies the model's logit probability landscape during training.

3 Methodology

We present our two QA datasets, followed by the train-time softmax tempering procedure with its extension via reinforcement learning to control temperature dynamically in Section 3.3. We also describe our training setup and evaluation criteria.

3.1 Datasets

Super SQuAD We modify SQuAD v2 (Rajpurkar et al., 2018), by changing both the input and output formats. For each sample, we concatenate between 1 to 500 random Wikipedia documents to create the context. Each sample has a context length of up to 32,000 tokens.

For each question, there are two possible cases: either the correct document is present in the context to answer the question, OR the correct document is NOT present. As was introduced in SQuAD v2, we want our model to recognize when the correct context is NOT present, and ideally avoid hallucinating an answer. In the other case, where the correct document CAN be found in the context, we want the model to copy it down the relevant paragraph to improve explainability and help auditing the system.

We set the labels to be multi-field JSONs. Specifically, if a question's document is present in the context, the model should return a JSON with four fields populated:

1. Question: The model copies the same question it was just asked.
2. Answer: The answer to the question.
3. Context: The model copies down the document in which the answer to the question is found.
4. Error code: 0 (This should always be 0 if the context and answer were found)

If the document is NOT in the context, the model is supposed to recognize this and should refuse to answer the question (even if it thinks it knows the answer). In this case, only the following two fields should be filled out in the resulting JSON:

1. Question: The model copies the same question it was just asked.
2. Error code: 1 (This should always be 1 when the model thinks the document needed to answer the question is not present)

Evaluation We evaluate using standard F_1, precision, and recall measures.

For cases where the "Correct Document Retrieved", then a test instance is marked:

- **True Positive**, if the Correct document is in the context AND the model copied it down correctly
- **False Positive**, if the Correct document is NOT in the context BUT the model copied something down
- **False Negative**, if the Correct document is in the context, BUT the model thought it wasn't there.
- **True Negative**, otherwise.

For cases where the "Model Correctly Realized Document is Missing", equivalently, we will have:

- **True Positive**, if the Correct document is NOT in the context, AND the model correctly put an error code of 1
- **False Positive** if the correct document is in the context, BUT the model incorrectly put an error code of 1
- **False Negative**, if the correct document is NOT in the context, BUT the model thought it was. It likely copied some other document.
- **True Negative**, otherwise.

PolyCompQA (Polymer Composite Question Answering) Dataset Recent studies have employed table question answering as a key approach to extract and utilize valuable information from tables found in various sources, including scientific papers (Jin et al., 2022; Ghosh et al., 2024). In materials science, several QA datasets exist, such as Battery Device QA (Huang and Cole, 2022), which includes 272 records for extractive question-answering on battery component classification; OpticalTable (Zhao et al., 2023), comprising $4,534$ tabular QA pairs; and MaScQA (Zaki et al., 2024), which covers general materials science QA, including inorganic composition extraction. To address the lack of domain-specific benchmarks for polymer composites, we curated PolyCompQA - a dataset derived from 207 scientific tables across 118 research papers - resulting in $4,270$ QA pairs. These papers were sourced from MaterialsMine (Brinson et al., 2020), a repository dedicated to polymer composite data. Each experimental sample in these tables is characterized by key attributes: the matrix (polymer), the filler material, its composition, and any applied particle surface treatments.

To ensure high-quality annotations, three domain-experienced researchers manually structured each table into lists containing:

- **Material sample composition descriptions** – capturing the matrix material, filler characteristics (including particle size (nano/micro), composition (wt% or vol%), and surface treatments).
- **Material properties** – listing the corresponding measured properties, such as glass transition temperature and tensile strength, for each sample along with the conditions under which they were measured.

Using these structured lists, we automatically generate question-answer pairs. Each question is formulated by constructing a detailed material description that integrates the polymer matrix, filler characteristics, and surface treatments, linking this description to a specific material property. When a property is measured under varying conditions (e.g., temperature or pressure), the relevant conditions are included in the question. An example table, along with the corresponding generated questions and answers, is illustrated in Figure 1. Tables exhibit significant diversity, as both sample and property information can be reported in various formats. In some cases, the entire sample description is condensed into a single cell, while in others, individual fields are spread across multiple columns. Property names may be explicitly stated in each row or column or only mentioned in the table title. Even for human annotators, distinguishing between the matrix, filler, and surface treatment can be challenging due to bespoke abbreviations. Additionally, incomplete sample descriptions often require cross-referencing other parts of the article. When key fields cannot be determined, they are annotated as "not specified" and excluded from question generation.

For each table, the maximum number of possible questions is determined by the number of experimental samples (n) multiplied by the number of reported properties (p), i.e., $n \times p$. For example, for the table in Figure 1, a total of $5\,(n) \times 3\,(p) = 15$ questions would be generated. Some properties are only available for specific samples. Extracting a complete set of samples and their corresponding properties presents a significant information extraction challenge (Gupta et al., 2022; Circi et al., 2024). We reformulated this task as a question-answering problem and obtained $4,270$ questions, where the sample details are provided in the question similarly to (Sipilä et al., 2024), and the LLM is expected to generate the correct property value. To convert article table images into text inputs, we use GPT-4o to generate a CSV format by prompting it with: "Please convert this table into a CSV format. The first row should contain the table title, and include all data and footnotes if present." Out of 369 unique properties, 364 of them occurred in $< 1\%$ of the questions. Total property distribution across all questions can be found in Figure 2.

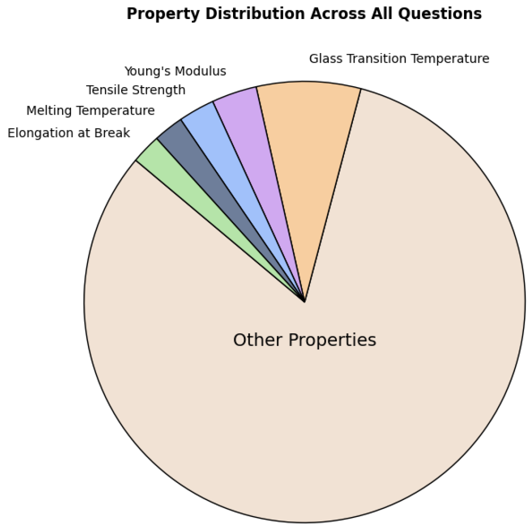

Figure 2: A pie chart showing the distribution of properties across all questions reveals that the most frequently reported material properties are: glass transition temperature (5.7%), young's modulus (2.5%), tensile strength (2.0%), melting temperature (1.6%), and elongation at break (1.6%).

3.2 Evaluation Metrics

PolyCompQA (Polymer Composite Question Answering) Task To effectively accelerate scientific discovery, LLMs must interact seamlessly with human scientists by answering questions reliably, making their evaluation and improvement essential (Miret and Krishnan, 2024). We calculate the accuracy score based on exact matching between the model's answers and ground truth labels. A binary score (1 for exact match, 0 for any mismatch) is assigned to each response. The final accuracy is calculated by dividing the total number of correct answers by the total number of questions.

3.3 Softmax Tempering

Following Dabre and Fujita (2020), who used Softmax Tempering to improve Machine Translation models, we let z represent the logits predicted by the model for the next token. The temperature T, which is a hyperparameter that controls the sharpness of the distribution, determines how much the logits z should be scaled before applying the softmax function. Dividing each logit z by the temperature T provides us with $\mathbf{z}_{\text{temp}}$ tempered version of the logits. Mathematically, the relationship is expressed as:

$$\mathbf{z}_{\text{temp}} = \frac{\mathbf{z}}{T}, \tag{1}$$

The temperature scaling effectively softens or sharpens the probability distribution depending on the value of T. The cross-entropy loss is then computed with respect to $\mathbf{z}_{\text{temp}}$. A higher temperature leads to a softer (more uniform) probability distribution, encouraging the model to avoid placing extremely high probability on a single token unless strongly justified by the context. A lower temperature makes the distribution sharper (more confident in fewer choices), penalizing the model for incorrect and noisy predictions. Although temperature scaling is typically only used at inference time, here it is integrated into the standard cross-entropy loss.

Later work by (Li et al., 2022) brought softmax tempering to the code generation domain. However, the authors reported that using values below 1.0 for T worked better than values above 1.0, which stands in contrast in the results in (Dabre and Fujita, 2020). While these works feature different goals - translation and code generation - we decide to investigate these different observations and perform a grid search over a wide variety of values for T.

Specifically, for PolyCompQA, we sweep these settings:

$$T \in \left\{ \begin{array}{l} 0.2,\ 0.36,\ 0.52,\ 0.68,\ 0.84, 1.0,\ 1.1, \\ 1.25,\ 1.43,\ 1.67,\ 2.0,\ 3.0 \end{array} \right\}$$

We finetune LLaMa-1B-Instruct for one epoch on our PolyCompQA dataset. We create a 10% validation for our hyperparameter sweep and report results on a 10% test split.

For Super SQuAD, which is significantly more costly to train on, we use these:

$$T \in \{\, 0.7,\ 0.8,\ 0.9,\ 1.0,\ 1.1,\ 1.2,\ 1.3, \}$$

.

For finetuning runs on both of these datasets, we also included three more hyperparameters:

- Weight decay: $\{0.0,\ 0.1\}$
- Learning rate: $\{5 \times 10^{-5},\ 5 \times 10^{-4}\}$
- Batch size: $\{32,\ 64\}$

All combinations were trained under the same random initialization and data order for a single epoch. For each configuration in the powerset, we report the best performing settings as measured on the test set of PolyCompQA.

When we train with Birdie, we keep these same settings in our hyperparameter sweep.

Universally, the best found settings were a batch size of 64, a learning rate of 5×10^{-4}, and a weight decay of 0.1.

3.4 Adaptive Softmax Tempering with Birdie

In addition to sweeping fixed softmax temperatures, we explore *dynamic* scheduling of the temperature T via Birdie (Blouir et al., 2024), an reinforcement learning-based method originally designed to mix different pre-training objectives automatically. Here, we adapt Birdie in two important ways:

1. Rather than minimizing per-objective *losses*, we replace Birdie reward function with one to maximize validation set accuracy. Specifically, we measure accuracy by generating responses to the test set every 256 steps. We define the scalar reward as

$$R \;=\; (accuracy_{new})^2 \;-\; (accuracy_{old})^2.$$

2. We use Birdie to ultimately predict a probability vector (of three dimensions) that correlate to three discrete softmax-temperature settings: $T \in \{0.5,\ 1.0,\ 1.67\}$. Each "action" that Birdie outputs is used by the trainer to choose a given setting given this probability vector.
3. Given enough time, the reward for a given set of actions as a signal allows Birdie to predict rewards given an action. Then, we can generate random actions to Birdie, and choose one with a high predicted reward to estimate which would be a good action. In this case, actions are the per-batch sampling ratios of different values of T, the softmax tempering hyperparameter.

Searching through the hyperparameter settings in subsection 3.3, we train Llama-1B-Instruct on the PolyCompQA dataset with Birdie controlling the softmax tempering hyperparameter T. Every 256 steps, we pass the previous validation set accuracy, the current accuracy, and the action probability vector that was taken. Birdie trains its internal model, a miniature decoder-only Transformer, and outputs calculates the reward R for each candidate action, and updates its internal reward model for 200 steps to pick the next probability distribution over $\{T = 0.5,\ T = 1.0,\ T = 1.67\}$. In practice, we sample 2,048 candidate distributions (actions), run them through Birdie's state model pick the action with the highest predicted reward to act as our new action distribution.

4 Experiments

4.1 PolyCompQA

We plot a chart of the best performances for each softmax temperature in Figure 3, and Table 2 shows the validation accuracy for the tested temperatures on both QA tasks. $T = 0.5$ and $T = 1.67$ each

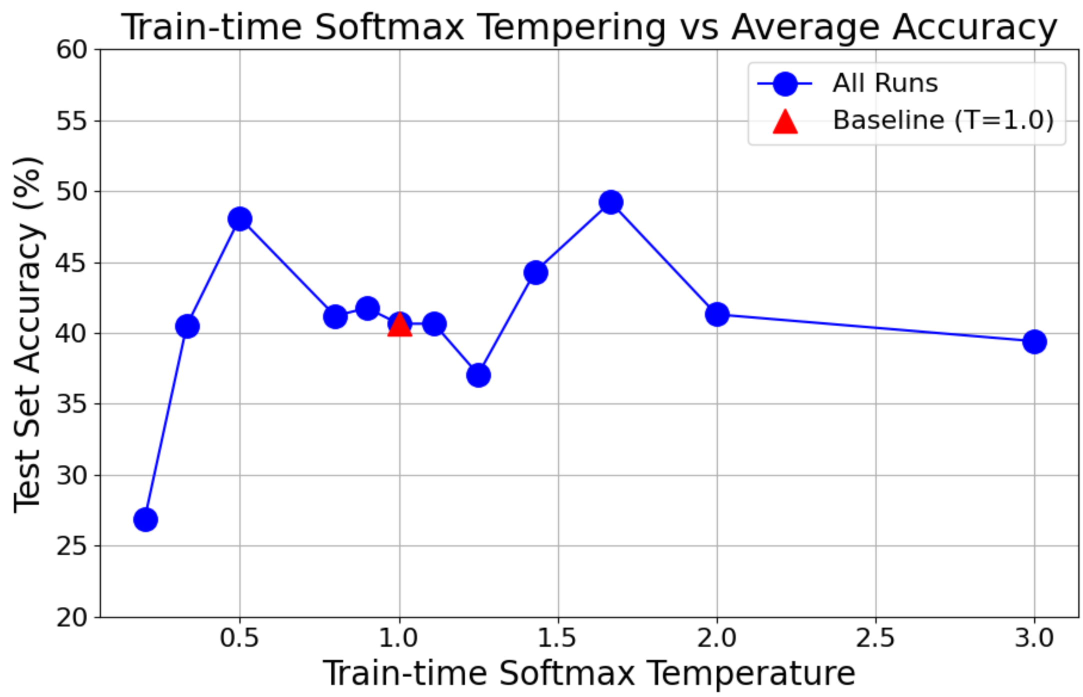

Figure 3: Train-time Softmax Temperature versus Test Set accuracy on our PolyCompQA dataset with a finetuned Llama-3.1-1B-Instruct. We see the standard softmax tempering setting of 1.0 underperform compared to T = 0.5 and T = (5/3). A "Cat Ear" shape emerges. Results are the maximum accuracy for each run over several hyperparameter searches and are described in subsection 3.3. Details on fine-tuning on this dataset are included in subsection 4.1.

outperform the default $T = 1.0$, reflecting the non-monotonic "cat-ear" pattern.

We also evaluate the GPT-4o, o3-mini, LLaMa 3.1 1B, and LLaMa 3.1 8B models on the 449 test set questions. The results are presented in Figure 4, and the prompts are detailed in Section 7.3. We observe that the GPT-4o model slightly outperforms the o3-mini model, with the GPT-4o achieving an accuracy of 0.686 and the o3-mini achieving 0.6459 in the zero-shot setting. There is no significant difference between the zero-shot and three-shot settings, as the GPT-4o model achieved an accuracy of 0.6704 in the three-shot setting, and the o3-mini model achieved 0.6437. The LLaMa 3.1 8B model achieved an accuracy of 0.509, while the LLaMa 3.1 1B model achieved 0.129.

4.2 Super SQuAD

In Table 2, we show results for Softmax Tempering on PolyCompQA. We see Birdie is is virtually tied with the best run.

Temperature	Noticing Context is Missing F_1 (%)	Retrieving Correct Context F_1 (%)
0.7	69.3	41.6
0.8	53.6	33.3
0.9	42.9	29.2
1.0	59.8	34.2
1.1	54.6	30.8
1.2	48.7	28.7
1.3	59.4	35.6
Base Llama-1B-Instruct	46.1	0.0

Table 1: F_1 scores (%) for 8,183 question–context pairs on Super SQuAD. Lower-temperature settings (e.g., $T = 0.7$) can help the model notice when the contxet is missing, as well as when retrieving the correct context. The base instruct model is unable to retrieve any context correctly. We include Llama-1B-Instruct before finetuning on Super SQuAD. More details on the task and finetuning procedure are available in section 3.1.

5 Discussion

5.0.1 Discussion of Temperature Trends

Our exploration of *train-time softmax tempering* during fine-tuning on both the expanded SQuAD v2 (*Super SQuAD*) and our newly created *PolyCompQA* dataset underscores the nuanced ways that manipulating the model's probability distribution at training time can improve performance in downstream QA tasks.

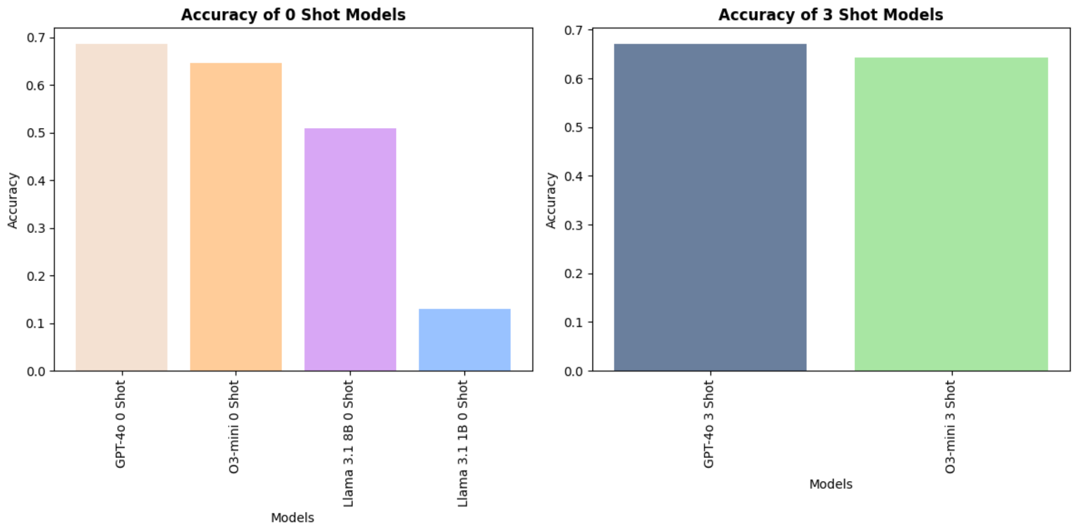

Figure 4: Accuracy results for different models in zero-shot and three-shot settings on the PolyCompQA dataset.

softmax_temperature	accuracy
0.20	26.8
0.330	40.5
0.50	48.1
0.80	41.2
0.90	41.7
1.00	40.6
1.11	40.7
1.25	37.1
1.43	44.3
1.67	**49.2**
2.00	41.3
3.00	39.4
Birdie	**49.4**

Table 2: Accuracy at different softmax temperatures

6 Conclusion

In information extraction and QA, we observe that even frontier language models can hallucination significantly. Through our investigation of softmax tempering and adaptive scheduling, we demonstrate reduced hallucination on two QA datasets, including our newly introduced PolyCompQA. Our benchmarking reveals that even large parameter models struggle with question answering on this specialized dataset. We improve the performance of the LLaMa 3.2 1B model through softmax tempering by using temperatures that deviate from the default value of 1.0 in both directions. Furthermore, reinforcement learning-based dynamic parameter adjustment during training shows promise in further enhancing model performance, suggesting a path forward for developing more reliable QA systems. We hope that future work looks closer at why both wildly different settings of softmax tempering - whether it is minimizing or maximizing entropy - can empirically help performance.

Limitations

Due to the diverse ways materials can be synthesized and processed in laboratory experiments, some samples in the PolyCompQA dataset cannot be fully distinguished using the predefined key fields. While these fields, identified by domain experts, are sufficient for differentiating most samples reported in the articles, certain cases require additional context - such as the presence of multiple fillers or variations in processing methods - that can significantly impact the material's properties. As a result, given a question based on a sample description, it may not always be possible to uniquely identify the corresponding material and, consequently, determine its true property value. Future work can address this limitation by incorporating more detailed annotations, including processing conditions, and generating questions that account for these factors. Our results with softmax tempering were performed on an academic budget and were exclusively performed on LLaMa 3.2 1B, which only has 1.2B parameters. Our results may not extrapolate to significantly larger models. Additionally, our experiments also deal with small finetuning datasets.

Ethical Considerations

We do not believe there are any ethical issues associated with this research.

References

Sam Blouir, Jimmy T.H. Smith, Antonios Anastasopoulos, and Amarda Shehu. 2024. Birdie: Advancing state space language modeling with dynamic mixtures of training objectives. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9679–9705, Miami, Florida, USA. Association for Computational Linguistics.

L Catherine Brinson, Michael Deagen, Wei Chen, James McCusker, Deborah L McGuinness, Linda S Schadler, Marc Palmeri, Umar Ghumman, Anqi Lin, and Bingyin Hu. 2020. Polymer nanocomposite data: curation, frameworks, access, and potential for discovery and design. *ACS Macro Letters*, 9(8):1086–1094.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:1877–1901.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Paull Hernandez, Margaret Coombe, Alicia Guo, Shixiang Shane Gu, Karishma Malkan, Daphne Luong, Johnny Soraker, Qifan Wang, Madeleine van Zuylen, Luke Culbertson, Suchin Gururangan, Edward J. Hu, Mia Chen, Ariel Hard, Khyati Mahajan, Nishant Subramani, Xiang Lisa Li, Yuxuan Cai, Zora Tung, Vincent Ying, Nan Du, YaGuang Li, Yanping Huang, Andrew M. Dai, Quoc Le, Zhifeng Chen, Claire Cui, Ed H. Chi, Ruoming Pang, Anthony P. F. Yu, Minhua Chen, Sunipa Dev, Maheep Gupta, Terry K. Hart, Julian T. Michael, Ryan P. Adams, Wei Han, Slav Petrov, Pengcheng Yin, Ben Taskar, Jeff Dean, and Slav Petrov. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint*, arXiv:2204.02311.

Defne Circi, Ghazal Khalighinejad, Anlan Chen, Bhuwan Dhingra, and L Catherine Brinson. 2024. How well do large language models understand tables in materials science? *Integrating Materials and Manufacturing Innovation*, 13(3):669–687.

Raj Dabre and Atsushi Fujita. 2020. Softmax tempering for training neural machine translation models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Emilio Dorigatti, Jann Goschenhofer, Benjamin Schubert, Mina Rezaei, and Bernd Bischl. 2024. Uncertainty-aware pseudo-label selection for positive-unlabeled learning. *Preprint*, arXiv:2201.13192.

Akash Ghosh, B Venkata Sahith, Niloy Ganguly, Pawan Goyal, and Mayank Singh. 2024. How robust are the tabular qa models for scientific tables? a study using customized dataset. *arXiv preprint arXiv:2404.00401*.

C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. 2017. On calibration of modern neural networks. *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS)*, 30:1321–1330.

Tanishq Gupta, Mohd Zaki, Devanshi Khatsuriya, Kausik Hira, NM Krishnan, et al. 2022. Discomat: distantly supervised composition extraction from tables in materials science articles. *arXiv preprint arXiv:2207.01079*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Shu Huang and Jacqueline M Cole. 2022. Batterybert: A pretrained language model for battery database enhancement. *Journal of chemical information and modeling*, 62(24):6365–6377.

Nengzheng Jin, Joanna Siebert, Dongfang Li, and Qingcai Chen. 2022. A survey on table question answering: recent advances. In *China Conference on Knowledge Graph and Semantic Computing*, pages 174–186. Springer.

S. Krishna, A. Gupta, R. Singh, and S. Agarwal. 2022. Re-ranking for improved question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1234–1245. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, and Sebastian Riedel. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:9459–9474.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. 2022. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3214–3252.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. Ambigqa: Answering ambiguous open-domain questions. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5783–5797.

Santiago Miret and NM Krishnan. 2024. Are llms ready for real-world materials discovery? *arXiv preprint arXiv:2402.05200*.

Samuel Müller and Andreas Müller. 2022. Temperature scaling for calibration in deep learning. *Journal of Machine Learning Research*, 23(1):1–26.

G. Pereyra, G. Tucker, J. Chorowski, Ł. Kaiser, and G. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 2880–2889. PMLR.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

M Sipilä, F Mehryary, S Pyysalo, F Ginter, and Milica Todorović. 2024. Question answering models for information extraction from perovskite materials science literature. *arXiv preprint arXiv:2405.15290*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Ronald J. Williams and David Zipser. 1989. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, 1(2):270–280.

Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. 2024. Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*.

Mohd Zaki, NM Anoop Krishnan, et al. 2024. Mascqa: investigating materials science knowledge of large language models. *Digital Discovery*, 3(2):313–327.

Jiuyang Zhao, Shu Huang, and Jacqueline M Cole. 2023. Opticalbert and opticaltable-sqa: text-and table-based language models for the optical-materials domain. *Journal of Chemical Information and Modeling*, 63(7):1961–1981.

7 Appendix

7.1 Hardware

All experiments were performed using a TPUv4-32 and four Nvidia H100 GPUs.

7.2 Super SQuAD Results

We show our detailed results in Table 3 and Table 4.

7.3 PolyCompQA Prompts

7.3.1 Zero-shot prompt

```
Given the table contents, answer the
    following question. Please provide
    only the numerical or categorical
    answer without any explanation. If
    the answer is a number, provide just
     the number without units.

{table_content}

Question: {question}
```

7.3.2 Three-shot prompt for PolyCompQA

We use these prompts for the few-shot models in Figure 4.

T	F_1 (%)	Prec (%)	Rec (%)	Acc (%)	FDR (%)	NPV (%)	FOR (%)	TPS	FPS	TNS	FNS	NPL	NNL	NPP	NNP	NP
0.7	41.6	59.7	32.0	43.8	40.3	35.9	64.1	1,639	1,108	1,949	3,487	5,126	3,057	4,406	3,777	8,183
0.8	33.3	69.1	22.0	45.0	30.9	39.0	61.0	1,126	504	2,553	4,000	5,126	3,057	2,565	5,618	8,183
0.9	29.2	74.7	18.2	44.9	25.3	39.5	60.5	932	316	2,741	4,194	5,126	3,057	1,802	6,381	8,183
1	34.2	63.8	23.4	43.7	36.2	37.7	62.3	1,197	679	2,378	3,929	5,126	3,057	3,154	5,029	8,183
1.1	30.8	65.4	20.1	43.3	34.6	38.0	62.0	1,032	546	2,511	4,094	5,126	3,057	2,675	5,508	8,183
1.2	28.7	70.1	18.1	43.8	29.9	38.8	61.2	927	396	2,661	4,199	5,126	3,057	2,174	6,009	8,183
1.3	35.6	65.1	24.5	44.5	34.9	38.1	61.9	1,256	674	2,383	3,870	5,126	3,057	3,127	5,056	8,183

Table 3: F1 score for retrieving the correct source document for a given question. The document is retrieved from a context with 1 - 500 random Wikipedia documents, if present. Results are from our finetuned Llama-3.1-1B-instruct. Please see section 3.1 for more details. FDR stands for false discovery rate, NPV for negative predictive value, FOR for false ommission rate, TPS for TP Sum, FPS for FP Sum, TNS for TN Sum, FNS for FN Sum, NPL for number of positive labels, NNL for number of negative labels, NPP for number of positive predictions, NNP for number of negative predictions, and NP for number of generations.

T	F_1 (%)	Prec (%)	Rec (%)	Acc (%)	FDR (%)	NPV (%)	FOR (%)	TPS	FPS	TNS	FNS	NPL	NNL	NPP	NNP	NP
0.7	69.3	74.9	64.4	64.2	25.1	51.7	48.3	3,301	1,105	1,952	1,825	5,126	3,057	4,406	3,777	8,183
0.8	53.6	80.4	40.2	56.4	19.6	45.4	54.6	2,061	504	2,553	3,065	5,126	3,057	2,565	5,618	8,183
0.9	42.9	82.5	29.0	51.7	17.5	43.0	57.0	1,486	316	2,741	3,640	5,126	3,057	1,802	6,381	8,183
1	59.8	78.5	48.3	59.3	21.5	47.3	52.7	2,476	678	2,379	2,650	5,126	3,057	3,154	5,029	8,183
1.1	54.6	79.6	41.5	56.7	20.4	45.6	54.4	2,129	546	2,511	2,997	5,126	3,057	2,675	5,508	8,183
1.2	48.7	81.8	34.7	54.2	18.2	44.3	55.7	1,778	396	2,661	3,348	5,126	3,057	2,174	6,009	8,183
1.3	59.4	78.4	47.9	59.1	21.6	47.1	52.9	2,453	674	2,383	2,673	5,126	3,057	3,127	5,056	8,183

Table 4: F1 score for correct error code predictions, where our finetuned Llama-3.1-1B-instruct outputs a 1 to indicate it cannot find the original source document of a question on our Super SQuAD dataset. Please see the task details in section 3.1. Train-Time Softmax Tempering. FDR stands for false discovery rate, NPV for negative predictive value, FOR for false ommission rate, TPS for TP Sum, FPS for FP Sum, TNS for TN Sum, FNS for FN Sum, NPL for number of positive labels, NNL for number of negative labels, NPP for number of positive predictions, NNP for number of negative predictions, and NP for numbr of predictions.

```
Given the table contents, answer the
    following question. Please provide
    only the numerical or categorical
    answer without any explanation. If
    the answer is a number, provide just
     the number without units. Do not
    include the standard deviation or
    any other information.

Examples:

Table: Table 1. Weibull parameters for
    Epoxy TiO2 composites.
Composition,Shape Parameter ,Scale
    Parameter
Unfilled,8.789,52.30
0.1% nano,11.11,33.28
0.5% nano,13.46,28.64
1% nano,10.17,33.71
5% nano,12.36,30.15
10% nano,8.154,34.57
10% micron,25.45,38.43

Question: What is the shape parameter
    value for epoxy with TiO2 filler at
    0.1%?

Answer: 11.11

Table: TABLE II: Dynamic Mechanical
    Properties of EVA and Its
    Nanocomposites
Sample,Tg (°C),E' (Pa) at Tg,E' (Pa) at
    30°C,tan δ at Tg,tan δ at 30°C
Pure EVA,-27,05 × 10^7,1.5 × 10^
    6,0.95,0.17
EVA + 4 wt % 12Me-MMT,-30,1.9 × 10^8,04
    × 10^6,0.68,0.16
EVA + 6 wt % 12Me-MMT,-32,06 × 10^8,07 ×
     10^6,0.55,0.17

Question: What is the storage modulus
    value at temperature 30 °C for EVA
    with no filler?

Answer: 1500000.0

Table: Table 1. Properties of UV cured
    samples.
Cured sample (a),Epoxy group conversion
    (b) %,Gel content (c) %,Tg (d) °C,
    Absorption maximum (e) nm
CE + 5 wt.-% AgSbF6,82,98,182,400
CE + 10 wt.-% AgSbF6,80,100,170,410
CE + 20 wt.-% AgSbF6,87,98,156,415
- (a) The cured sample composition.
- (b) Epoxy group conversion percentage.
- (c) Gel content percentage.
- (d) Glass transition temperature.
- (e) Absorption maximum wavelength.

 Question: What is the glass transition
     temperature value for CE with
     AgSbF6 filler at 20%?
Answer: 156

Table: {table_content}
Question: {question}
Answer:
```

7.4 Super SQuAD

7.5 Risks

This could help material science progress and help nefarious individuals make bad items.

```
Super SQuAD Sample:

[[1-500 Articles Total]]

In 1907, the newly established Board of Education found that greater
capacity for higher technical education was needed and a proposal
to merge the City and Guilds College, the Royal School of Mines and
the Royal College of Science was approved and passed, creating
The Imperial College of Science and Technology ...

A strong consistent theme in his family-friendly work is a childlike,
even naïve, sense of wonder and faith, as attested by works such
as Close Encounters of the Third Kind, E.T. the Extra-Terrestrial, ...

Relatively insensitive film, with a correspondingly lower speed
index, requires more exposure to light to produce the same image
density as a more sensitive film, and is thus commonly termed
a slow film. ...

What speed of film is produced by insensitive film?
Please format your response using JSON.
```

```
{
    "question": "What speed of film is produced by insensitive film?",
    "error_code": 0,
    "context": "Relatively insensitive film, with a correspondingly lower speed index, requires more
   exposure to light to produce the same image density as a more sensitive film, and is thus commonly termed
    a slow film.",
}
```

Figure 5: A sample from our Super SQuAD dataset, which is a remix of SQuAD-v2 dataset. We concatenate hundreds of Wikipedia documents together. The question requires identifying if the original source document can be found in the context. We concatenate random articles together until we reach a target length of 32,000 tokens. More details are available in section 3.1.