Hypothesis Generation with Large Language Models

Yangqiaoyu Zhou[♣], **Haokun Liu**[♣], **Tejes Srivastava**[♣]
Hongyuan Mei[†] & **Chenhao Tan**[♣]
Department of Computer Science
University of Chicago[♣], Toyota Technological Institute at Chicago[†]
Chicago, IL 60637, USA
{zhouy1,haokunliu,tejess,chenhao}@uchicago.edu,hongyuan@ttic.edu

Abstract

Effective generation of novel hypotheses is instrumental to scientific progress. So far, researchers have been the main powerhouse behind hypothesis generation by painstaking data analysis and thinking (also known as the Eureka moment). In this paper, we examine the potential of large language models (LLMs) to generate hypotheses. We focus on hypothesis generation based on data (i.e., labeled examples). To enable LLMs to handle long contexts, we generate initial hypotheses from a small number of examples and then update them iteratively to improve the quality of hypotheses. Inspired by multi-armed bandits, we design a reward function to inform the exploitation-exploration tradeoff in the update process. Our algorithm is able to generate hypotheses that enable much better predictive performance than few-shot prompting in classification tasks, improving accuracy by 31.7% on a synthetic dataset and by 13.9%, 3.3% and, 24.9% on three real-world datasets. We also outperform supervised learning by 12.1% and 11.6% on two challenging real-world datasets. Furthermore, we find that the generated hypotheses not only corroborate human-verified theories but also uncover new insights for the tasks.

1 Introduction

Hypothesis generation drives scientific progress. Mendel's hypothesis on allele pairs lays the foundation for modern genetics; Einstein's hypothesis in general theory of relativity led to the prediction and subsequent confirmation of gravitational waves. In the context of language modeling, the hypothesis on scaling law inspires recent progress in large language models (LLMs) (Kaplan et al., 2020). Despite the importance of hypothesis generation, as Ludwig and Mullainathan (2024) point out, science has been curiously asymmetric. While many scientific publications present extensive formal and empirical evaluation of hypotheses, the generation of hypotheses happens off-stage by researchers. In order to generate novel hypotheses, researchers may read literature, analyze data, pick the brain of each other, and even "hallucinate" (see Kekulé's discovery of the structure of the benzene molecule (Rothenberg, 1995)). Given the rise of large language models (Brown et al., 2020; Anthropic, 2023; OpenAI, 2023b), we examine their potential of providing much needed assistance in hypothesis generation in this work.

In particular, we focus on hypothesis generation based on data, a common approach in empirical sciences. Our main question is how we can enable LLMs to generate hypotheses of high-quality. While one can easily prompt LLMs to generate hypotheses, LLMs may not be able to effectively leverage the input examples in a single long prompt. Moreover, it is important to have measures of quality in the generation process so that we can filter bad hypotheses and come up with better ones. These two observations motivate us to start with a setup analogous to supervised learning. We can iteratively prompt an LLM to generate hypotheses based on the training examples and use training accuracy as a measure of quality to guide the generation process. Conveniently, we can also evaluate the quality of the final generated hypotheses with their performance on held-out examples, similar to supervised learning.

To generate high-quality hypotheses with LLMs, we propose an algorithm inspired by the upper confidence bound algorithm in multi-armed bandits (Auer, 2002) (**HypoGeniC**[1], **Hypo**thesis **Gen**eration **i**n **C**ontext; see Figure 1). Given initial hypotheses generated from a small number of examples, we need to assess their quality and propose new hypotheses to address their deficiencies. To navigate this exploration-exploitation tradeoff, we introduce a reward function and evaluate the top k hypotheses for each training example. We maintain a wrong example bank to capture the gap in knowledge of the hypotheses pool, and generate new hypotheses based on the wrong example bank to close the gap.

The generated hypotheses naturally enable an interpretable hypothesis-based classifier. We propose a suite of inference strategies given a set of hypotheses. We apply our method to one synthetic task where there is a single known valid hypothesis and three real-world tasks (DECEPTIVE REVIEWS, HEADLINE POPULARITY, and TWEET POPULARITY). The real-world tasks focus on deception detection and message popularity prediction, which are known to be challenging even for humans (Ott et al., 2011; Salganik et al., 2006). Our al-

[1]We have publicly released the code and data for **HypoGeniC** at https://github.com/ChicagoHAI/hypothesis-generation.

Proceedings of the 1st Workshop on NLP for Science (NLP4Science), pages 117–139

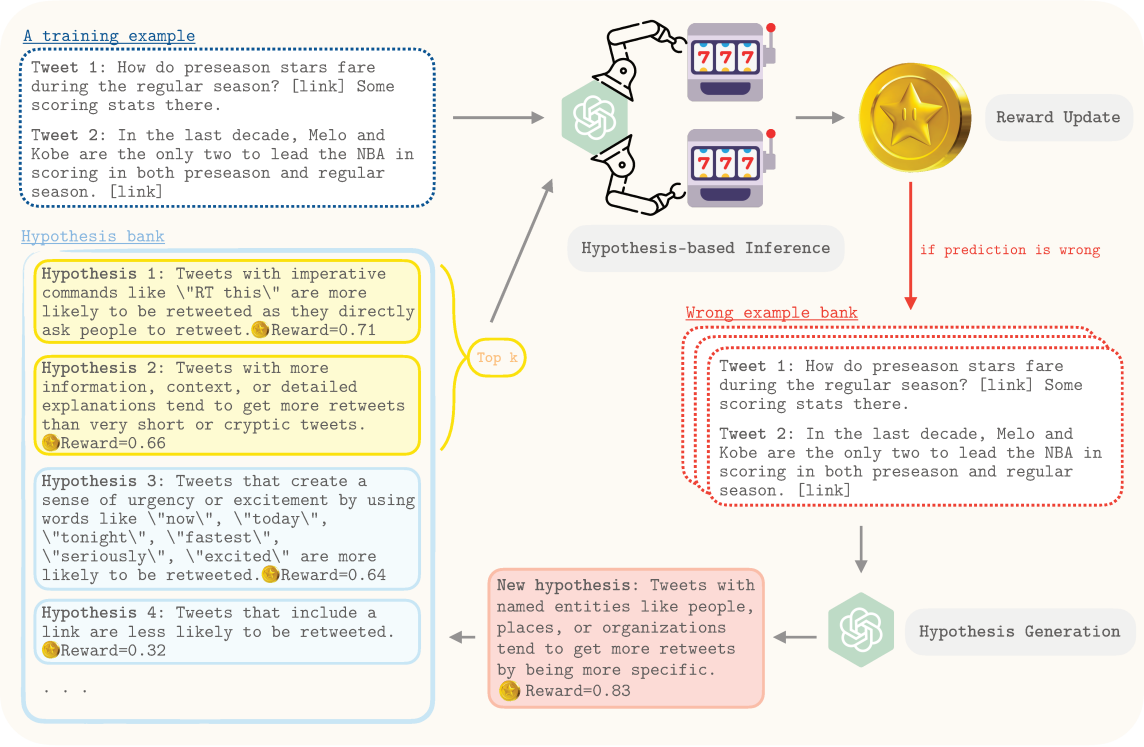


Figure 1: Illustration of **HypoGeniC**. During update stage, we evaluate the top k hypotheses on each new training example and update the reward based on the prediction correctness. If the number of hypotheses that got the example wrong exceeds a certain threshold, we add the example to a wrong example bank. The wrong example bank is then used to generate new hypotheses.

gorithm can recover the hypothesis in the synthetic task and also provide useful hypotheses for the real-world tasks. In fact, our generated hypotheses consistently outperform few-shot in-context learning baselines across all four tasks (31.7% in SHOE SALES, 13.9% in DECEPTIVE REVIEWS, 3.3% in HEADLINE POPULARITY, and 24.9% in TWEET POPULARITY). The predictive performance matches and even outperforms oracle supervised learning with RoBERTa and Llama-2-7B except in DECEPTIVE REVIEWS.

It is important to emphasize that although the utility of hypotheses in assisting downstream classification serves as an indicator for LLMs' ability to generate hypotheses, **our goal is not to maximize the classification performance.** Rather, our primary interest lies in the **quality of the hypotheses**. Thus, it is critical for the hypotheses to be interpretable beyond the LLM used to produce the hypotheses. We show that hypotheses generated by one LLM (e.g., GPT-3.5-turbo) can be used to make accurate inference by another LLM (e.g., Mixtral). On an out-of-distribution dataset for DECEPTIVE REVIEWS, we can even outperform the oracle fine-tuned RoBERTa. Such cross generalization provides strong evidence that we are able to generate hypotheses of high quality. Furthermore, through a qualitative analysis, **our generated hypotheses not only confirm theories from existing literature but also provide new insights about the task.** For instance, one novel hypothesis is that "reviews that mention personal experiences or special occasions, such as birthdays, anniversaries, or weddings, are more likely to be truthful". We encourage future research on deception detection to explore these novel hypotheses.

To summarize, we make the following contributions:

- We propose a novel computational framework for generating and evaluating hypotheses with LLMs.
- Our generated hypotheses enable interpretable hypothesis-based classifiers that outperform in-context learning and even supervised learning for one synthetic and three real-world datasets. These hypotheses are also robust across different LLMs and out-of-distribution datasets.
- Our generated hypotheses corroborate existing findings while also providing new insights for the tasks.

2 Method

We begin with a description of the problem formulation. Given a set $\mathcal{S} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where x_i is an example and y_i is the corresponding label, the goal is to learn a set of hypotheses $\mathcal{H} = \{h_1, ..., h_m\}$ that describe theories of relationships between x and y. To this end, we prompt an LLM to summarize demonstration examples into high-level hypotheses (§ 2.1). Then, during inference, the LLM makes inference based on the generated hypothesis (§ 2.2).

2.1 Hypothesis Generation

Our hypothesis generation algorithm (Algorithm 1) is inspired by the upper confidence bound (UCB) algorithm (Auer, 2002). Given a set of initial examples $\mathcal{S}_{\text{init}} \subset \mathcal{S}$, we first prompt an LLM to generate hypotheses for $\mathcal{S}_{\text{init}}$, which serve as our initial hypothesis bank $\mathcal{H}$. While initialized hypotheses may explain some portions of data, they often fall short of encompassing the full scope of the examples. We thus introduce an update stage which serves a dual purpose: 1) it increases the percentage of data explainable by the hypotheses and 2) it replaces any hypotheses that are found to be inaccurate.

In the update stage, for a training example s, we select the top k high-reward hypotheses from the hypothesis bank $\mathcal{H}$. The LLM is prompted to make a prediction with each of the top k high-reward hypotheses on s. Then we compute the accuracy of the inference and accordingly update the reward for each of the hypotheses. If w_{hyp} hypotheses predict incorrectly for the example s, then s is added to a wrong example pool $\mathcal{W}$. Once the wrong example pool reaches a max size of w_{max}, the wrong examples in $\mathcal{W}$ are used to generate new hypotheses. The wrong example pool represents the gap in knowledge that the current pool of hypotheses has for the dataset. Thus, by generating new hypotheses, the algorithm fills in these gaps. We update $\mathcal{H}$ with the newly generated hypotheses as per the rewards.

Reward. As mentioned above, each hypothesis has an associated reward. In our algorithm, we use the reward function in the UCB algorithm due to similarities between the multi-arm bandit problem and our problem formulation. In particular, we consider each hypothesis to be an arm and each training example to be a "pull". We note, however, that unlike the multi-arm bandit problem, multiple hypotheses are tested for a singular train example. Moreover, there can be new arms after hypotheses are updated, altering the setting from the standard static arms scenario to a dynamic arms scenario. Formally, the reward is defined as

$$r_i = \frac{\sum_{(x_j,y_j)\in\mathcal{S}_i} I(y_j = \hat{y}_j)}{|\mathcal{S}_i|} + \alpha\sqrt{\frac{\log t}{|\mathcal{S}_i|}}, \quad (1)$$

where $\mathcal{S}_i$ is the set of examples that have been used to evaluate the hypothesis h_i, t is train time step, and α is a hyperparameter that controls the exploration term. The first term in the reward function denotes the accuracy of the hypothesis for all $\mathcal{S}_i$. The second term is the exploration term, which is computed based on the number of times the hypothesis has been selected and the number of training examples visited so far. The accuracy term urges the algorithm to use well-performing hypotheses, whereas the exploration term encourages the algorithm to explore hypotheses that have not been selected many times. Thus, the reward function strikes a balance between exploration and exploitation.

For more details on implementation of **HypoGeniC**, refer to Appendix B.1.

Algorithm 1 HypoGeniC

Input: Training samples $\mathcal{S}$, `num_init`, k, w_{max}, H

// *Initialize hypothesis bank*
$\mathcal{H} \leftarrow$ **generate_hypotheses**($\{\mathcal{S}_i : i \leq$ `num_init`$\}$)
$\mathcal{W} \leftarrow \{\}$
for $(x_t, y_t) \in \mathcal{S}$:
  $\mathcal{H}_{\text{top}} \leftarrow \{h : h \in \mathcal{H}$ has top k reward$\}$
  for $h \in \mathcal{H}_{\text{top}}$:
    $\hat{y}_t^h \leftarrow$ **inference**(h, t)
    update_reward$(h, y_t, \hat{y}_t^h)$
  if $|\{\textbf{wrong}(\hat{y}_t^h) : h \in \mathcal{H}\}| \geq w_{hyp}$:
    // w_{hyp} *is dynamically determined, see Appendix B.1*
    $\mathcal{W} \leftarrow \mathcal{W} \cup \{(x_t, y_t)\}$
  if $|\mathcal{W}| = w_{max}$:
    $\mathcal{N} \leftarrow$ **generate_hypotheses**$(\mathcal{W})$
    $\mathcal{W} \leftarrow \{\}$
    $\mathcal{H} \leftarrow \{h : h \in \mathcal{H} \cup \mathcal{N}$ has top k reward$\}$
return $\mathcal{H}$

2.2 Hypothesis-based Inference

For efficiency purposes, we use each hypothesis on its own without accounting for their combinatorial effect during training; however, we should leverage the set of hypotheses as a whole during inference for at least two reasons. Firstly, some hypotheses may only apply to a subset of examples. Second, competing theories may require head-to-head comparisons. Hence, we develop multiple inference strategies to account for these different styles of reasoning (see Appendix A for prompts and Appendix B.2 for implementation details).

- **Best-accuracy hypothesis.** The hypothesis h with the highest accuracy from the hypothesis bank is included in the prompt to guide the model to perform inference.

- **Filter and weighted vote.** One hypothesis may not be enough to explain the data. Thus, this approach uses a combination of relevant hypotheses to make predictions for a single example. We first *filter* hypotheses by prompting an LLM to judge which hypotheses are relevant to the example. Next, an LLM is prompted to generate predictions for each of the relevant hypotheses, and these predictions are aggregated with *weighted vote*, where the weight is the training accuracy of the corresponding hypothesis.

- **Single-step adaptive inference.** Similar to *filter and weighted vote*, this approach leverages contextual information to choose hypotheses. The difference, however, is that it selects the most applicable hypothesis for each test example. Specifically, for a given test example, the LLM is tasked with identifying the most applicable hypothesis from a set of options. For each hypothesis, we provide instances from the training set where the hypothesis was accurate. Then, the LLM selects the most relevant hypothesis by comparing the test example to these training examples and evaluating their similarity. Thereafter, we apply the hypothesis to the test ex-

ample to perform inference. Please note that this is all done in one step with a long prompt.

- **Two-step adaptive inference.** We divide the previous inference strategy into two steps:
  1. The LLM determines the most relevant set of examples by comparing the test example with the corresponding examples of the hypotheses.
  2. Then, the corresponding hypothesis is provided to the LLM, which it uses to perform inference on the test example in a second prompt.

3 Experiment Setup

We introduce the experiment setup to evaluate **HypoGeniC**.

3.1 Tasks and Datasets

The choice of appropriate tasks is critical for evaluating the ability of LLMs to generate hypothesis. The focus of our work is on generating hypotheses based on observed data. A prerequisite is that potential hypotheses do exist. In the context of classification, it implies that the classification performance is non-trivial. In addition, we need to ensure that the hypotheses describing the data are likely not a priori known by LLMs, which rules out standard tasks such as sentiment analysis. Therefore, we use four datasets that satisfy these requirements: a synthetic task with a known true hypothesis and three *real-world* datasets that exhibit complex underlying patterns and constitute widely studied social science problems.

SHOE SALES is a synthetic task we created to investigate the scenario where there is only one single valid hypothesis. The task is to predict the color of the shoe that the customer will buy based on their appearance. The input provides appearance features, namely, age, height, gender, color of the hat, color of the shirt, color of the bag, and size of the bag. We construct this dataset such that the color of the shoe must match the color of the shirt. Since there are six colors in total, this becomes a 6-class classification problem.

Deceptive review detection is an instance of deception detection, a widely studied phenomenon in psychology and other social sciences (Granhag and Vrij, 2005). This particular task (DECEPTIVE REVIEWS) requires distinguishing genuine reviews from fictitious ones (Ott et al., 2011), where human performance is about chance (Lai and Tan, 2019). The dataset includes 800 genuine reviews and 800 fictitious reviews for 20 hotels in Chicago.

Predicting popularity is a notoriously challenging task in social sciences because it is known to be affected by seemingly random factors (Salganik et al., 2006). We use two datasets in this work: HEADLINE POPULARITY and TWEET POPULARITY. HEADLINE POPULARITY is derived from a dataset in the Upworthy Research Archive (Matias et al., 2021). The original dataset was collected through A/B testing, where each user was shown pairs of a headline and image for multiple packages (articles). Each user was exposed to only one of these pairs per package, and the clicks were recorded for each pair per package.[2] This process resulted in a total of 150,816 headlines across 22,666 packages. We construct a binary classification dataset by choosing the headlines that received the most clicks and least clicks for each package. We remove all sets of duplicate headlines, which results in our version of the HEADLINE POPULARITY dataset. The task for this dataset is to deduce which headline had more clicks in a pair. TWEET POPULARITY uses a dataset of 13,174 tweet pairs (Tan et al., 2014), which are matched by the topic and the author. Similar to HEADLINE POPULARITY, the task is to predict which one received more retweets.

3.2 Baselines, Oracles, and Evaluation Metrics

We use three different LLMs in our experiments (Mixtral (Mistral, 2023), GPT-3.5-turbo (OpenAI, 2023a), and Claude-2.1 (Anthropic, 2023)). We compare our approach with the following methods.

1. **Zero-shot and few-shot prompting.** We provide LLMs with task-specific instructions (zero-shot), optionally accompanied by three demonstration examples (few-shot).

2. **No updates.** To assess the value of the update stage in our algorithm, we evaluate the performance of the initialized hypotheses. In particular, we pick the best-performing hypothesis on the training set and use it for inference on the test set.

3. **Supervised Learning.** We fine-tune RoBERTa (Liu et al., 2019) and Llama-2-7B (Touvron et al., 2023) on each of the datasets to serve as a non-interpretable oracle. We include results for training on 200 examples and 1000 examples. Since fine-tuning update model weights, we expect RoBERTa and Llama-2-7B to set the upper bound on in-distribution datasets.

We randomly sample 200 training examples and 300 test examples for each dataset. Since all our datasets are classification tasks with ground truth labels, we use accuracy as our evaluation metric. To understand the effect of the number of training examples, we evaluate the performance of all methods at 10, 25, 50, 100, and 200 training examples. We also experiment with two different hypothesis bank sizes: 3 and 20 hypotheses to evaluate the impact of utilizing a larger number of hypotheses. The detailed hyperparameters of our approach can be found in Appendix B.3.

4 Results

To demonstrate the effectiveness of our hypothesis generation approach, we present results via three evaluation methods. First, we show that in the standard supervised

[2]The Upworthy Research Archive only provides the image IDs instead of the graphics. We thus only use the headlines for our dataset.

learning setup, our generated hypotheses enable more accurate predictions than baselines and even oracles when using a small set of examples. Second, we evaluate the generated hypotheses by checking whether they can generalize across different inference LLMs and to out-of-distribution datasets. We find surprisingly consistent performance even when using a different LLM to make inference from the generated hypotheses. So, we conduct a qualitative analysis to show that the generated hypotheses not only corroborate existing theories but also provide novel insights about the tasks at hand.

4.1 Performance on Heldout Test Sets

As discussed in the introduction, a side product of our approach is an interpretable hypothesis-based classifier. We compare its performance with standard supervised learning with the fine-tuned models and few-shot in-context learning (Table 1).

Our generated hypotheses improve inference over standard zero-shot and few-shot inference. Across all LLMs, **HypoGeniC** outperforms the zero-shot learning by an average of 60% on SHOE SALES, 22.7% on DECEPTIVE REVIEWS, 5.1% on HEADLINE POPULARITY, and 30.6% on TWEET POPULARITY. Similarly, we find that **HypoGeniC** shows an increase from few-shot learning by 31.7% on SHOE SALES, 13.9% on DECEPTIVE REVIEWS, 3.3% on HEADLINE POPULARITY, and 24.9% on TWEET POPULARITY. Note that these results are inflated on TWEET POPULARITY as safety mode is triggered for Mixtral and Claude-2.1 for zero-shot and few-shot learning respectively. After computing the 95% confidence intervals (with a binomial distribution assumption) for our results, the following results are significant for the real life datasets: **HypoGeniC** for DECEPTIVE REVIEWS and TWEET POPULARITY with Claude-2.1 and Mixtral, when comparing to their respective few shot baselines. If we relax the confidence interval to 90%, the result for HEADLINE POPULARITY with Mixtral is also statistically significant. These results demonstrate that hypothesis-based inference can increase the performance of LLMs significantly. Further results can be found in Table 5. One exception is that our method performs slightly worse (by 1%) than the few-shot baseline in the TWEET POPULARITY with GPT-3.5-turbo. One possible reason is that the few-shot demonstrations are effective at eliciting the pretraining knowledge in GPT-3.5-turbo, possibly due to a large amount of tweets in pretraining data. More detailed results are in Appendix C.

We also evaluate generated hypotheses with oracle inference, where the model retrospectively picks the best hypothesis for each prediction from the bank. With oracle inference, **HypoGeniC** achieves on average 88.6% on DECEPTIVE REVIEWS, 84.1% on HEADLINE POPULARITY, and 88% on TWEET POPULARITY across all LLMs, which are superior to results in Table 1. This result further suggests that hypotheses generated by **HypoGeniC** are of high quality and can lead to accurate predictions when the correct hypothesis is selected.

HypoGeniC matches or even exceeds the fine-tuned models with the same number of training examples on most datasets. Both **HypoGeniC** and the fine-tuned models yield 100% on the syntheic dataset. Moreover, **HypoGeniC** is 12.8% and 11.2% better than RoBERTa, and 12.1% and 11.6% better than Llama-2-7B, on HEADLINE POPULARITY and TWEET POPULARITY respectively with 200 training examples. Since the fine-tuned models learns by updating model weights to minimize the cross-entropy loss, it tends to benefit from more training examples, so we increase training examples to 1000 for the fine-tuned models. Despite the accuracy boost from more training examples, we find that **HypoGeniC**'s best result still outperforms RoBERTa by 3.7% and 0.7%, and Llama-2-7B by 3.7% and 11.4%, on HEADLINE POPULARITY and TWEET POPULARITY, respectively. One exception, however, is the DECEPTIVE REVIEWS dataset. We suspect that as word-level features are very useful in this dataset (Ott et al., 2011), they could be tougher for LLMs to extract but easier for fine-tuned models to grasp.

Updating the hypothesis bank leads to hypotheses of higher quality. Comparing **HypoGeniC** with the "no updates" results, we find that updating hypotheses generally leads to better hypotheses, suggesting that our algorithm is effective at improving hypothesis quality. The improvement is on average 0.7% on SHOE SALES, 5.8% on DECEPTIVE REVIEWS, 8.1% on HEADLINE POPULARITY, and 7% on TWEET POPULARITY. Another advantage of **HypoGeniC** over "no updates" is that sometimes the training examples exceed the context window size of LLMs, which can lead to degraded performance (Figures 4 and 5).

Effect of inference strategy. Figure 2 shows **HypoGeniC** results with different inference strategies on DECEPTIVE REVIEWS. Single-step adaptive inference is the most effective. Generally, we find hypotheses to be one-sided, focusing on either characteristics of truthful or deceptive reviews. We thus need to consider more than one hypothesis to make a correct prediction, so best-accuracy hypothesis or two-step adaptive inference are not ideal. On the other datasets, we find that the effect of inference strategy is much smaller (Figure 3). Best-accuracy hypothesis is sufficient for SHOE SALES and HEADLINE POPULARITY, and filter and weighted vote works best for TWEET POPULARITY. **Whichever inference strategy we use, the trend of HypoGeniC against few-shot learning and the fine-tuned models remains largely the same.**

Generally, having more training examples and a larger hypothesis pool improves performance. We show performance for different methods as number of training examples increase in Figures 4–6. We find **HypoGeniC** accuracy steadily increases as training size increases on SHOE SALES, suggesting that an LLM is

Models	Methods	SHOE SALES	DECEPTIVE REVIEWS	HEADLINE POPULARITY	TWEET POPULARITY
RoBERTa (Oracle)	Train 200	100.0	84.0	49.0	50.7
	Train 1000	100.0	91.0	60.0	62.0
Llama-2-7B (Oracle)	Train 200	100.0	88.7	49.7	50.3
	Train 1000	100.0	92.3	60.0	51.3
Claude-2.1	Zero shot	36.0	31.0	59.0	50.3
	Few shot	75.0	51.0	60.0	0.3*
	HypoGeniC (no updates)	100.0	70.3	57.3	59.0
	HypoGeniC	**100.0**	**75.3**	**61.3**	**62.0**
Mixtral	Zero shot	43.0	55.0	55.0	2.7*
	Few shot	79.0	56.3	55.3	48.7
	HypoGeniC (no updates)	96.0	60.3	59.7	60.7
	HypoGeniC	**98.0**	**68.0**	**60.3**	**62.7**
GPT-3.5-turbo	Zero shot	39.0	50.0	56.0	41.0
	Few shot	49.0	55.0	60.0	**62.0**
	HypoGeniC (no updates)	100.0	56.0	44.0	45.0
	HypoGeniC	**100.0**	**60.7**	**63.7**	61.0

Table 1: Prediction accuracies with 200 examples. We report the best numbers across all hyperparameter configurations, number of training examples, and inference strategies for **HypoGeniC** (we discuss their effect in details in § 4.1). The sensitive nature of the TWEET POPULARITY dataset may cause models to have their safety mode triggered. These results are marked by * in the table.

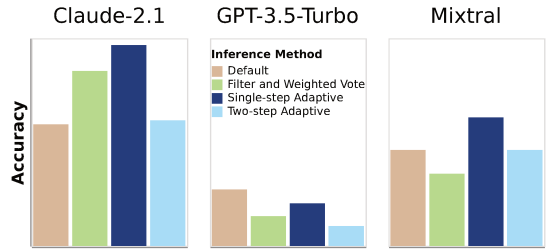


Figure 2: **HypoGeniC** results with different inference strategies on DECEPTIVE REVIEWS. Single-step adaptive hypothesis-based inference is generally the most effective on this dataset.

more likely to generate the best hypothesis given more examples. For the real-world datasets, however, the performance sometimes peaks at training size at 25 or 100 before reaching to 200. We suspect that the evaluation of the hypothesis bank would be less stable for the real-world datasets, since more than one correct hypotheses are needed for the task. We also find that using a hypothesis pool of size 20 leads to better performance than using a pool of size 3.

Although this classification experiment is convenient to run and demonstrates that our generated hypotheses are reasonable, our main goal is to generate high-quality hypotheses rather than maximizing the performance of this particular way of using the hypotheses. The next two experiments are essential in understanding the quality of hypotheses through generalization and manual analysis.

4.2 Generalization of the Generated Hypotheses

Our primary interest lies in the quality of the hypotheses. A good hypothesis should enable accurate inference by any AI model or even human and also generalize to unseen out-of-distribution dataset. In this subsection, we mix and match different LLMs for generation and inference. We also evaluate the hypotheses in deceptive review prediction on a new out-of-distribution (OOD) dataset (Li et al., 2013).

We find that the hypotheses generated by HypoGeniC generalize across models (Table 2). Generally, we find Claude-2.1 and Mixtral to be better at inference. Thus, substituting the inference model with them lead to better performance for hypothesis generated with GPT-3.5-turbo. Substituting Claude-2.1 and Mixtral as each other's inference model lead to small changes in performance. On SHOE SALES, the performance remains high for any inference model used.

Performance even increases for DECEPTIVE REVIEWS and HEADLINE POPULARITY when using Claude-2.1 as the inference model. For the cases where performance drops from Claude-2.1 to Mixtral, the decrease is marginal: 2.3% on DECEPTIVE REVIEWS and 2.7% on TWEET POPULARITY.

These results suggest that the hypotheses generated by **HypoGeniC** are generalizable across different LLMs, which somewhat contradicts the claim in Qiu et al. (2024) that LLMs cannot reliably interpret the hypotheses. We suspect that the reason is that our tasks only rely on natural language, while their tasks rely on notions of worlds and are fed into symbolic interpreters.

Generation Model	Inference Methods	SHOE SALES	DECEPTIVE REVIEWS	HEADLINE POPULARITY	TWEET POPULARITY
Claude-2.1	Claude-2.1	100.0	67.3	57.7	62.0
	Mixtral	94.0	65.0	57.7	59.3
	GPT-3.5-turbo	100.0	60.7	56.3	57.7
Mixtral	Claude-2.1	99.0	69.7	59.0	58.7
	Mixtral	98.0	61.3	57.7	59.3
	GPT-3.5-turbo	90.0	56.7	55.3	53.0
GPT-3.5-turbo	Claude-2.1	100.0	75.3	60.3	59.0
	Mixtral	98.0	62.0	60.0	62.3
	GPT-3.5-turbo	100.0	57.3	58.7	56.3

Table 2: Performance of cross-model generation and inference with train size = 200 using best-accuracy hypothesis inference and the best hypothesis bank size between 3 and 20.

Models	OOD
RoBERTa (Oracle)	73.0 (↓11.0)
Llama-2-7B (Oracle)	78.7 (↓10.0)
Claude-2.1 Few shot	41.7 (↓9.3)
Claude-2.1 **HypoGeniC**	**74.7** (↑4.7)
Mixtral Few shot	49.0 (↓7.3)
Mixtral **HypoGeniC**	64.7 (↑1.7)
GPT-3.5-turbo Few shot	52.0 (↓3.0)
GPT-3.5-turbo **HypoGeniC**	60.7 (↑3.4)

Table 3: Performance on OOD deceptive reviews.

Our generated hypotheses generalize to an out-of-distribution dataset. Table 3 presents an overview for the OOD deceptive review dataset. This dataset differs from DECEPTIVE REVIEWS by including reviews from four cities sourced from different websites (Li et al., 2013). We find that **HypoGeniC** outperforms few-shot learning by an average of 19.1%. Despite the distribution shift, **HypoGeniC** surprisingly increases accuracy from DECEPTIVE REVIEWS by an average of 3.3%, suggesting our hypotheses generalize well to this OOD dataset. Claude-2.1 remains the best performing model. In comparison, the performance of RoBERTa drops by 11%, and Llama-2-7B drops by 10%. As a result, **HypoGeniC** with Claude-2.1 outperforms RoBERTa by 1.7%, demonstrating the robustness of hypothesis-based inference. Refer to Appendix C.3 for more details.

4.3 Qualitative Analysis

For the synthetic dataset, all models are able to find the true underlying hypothesis for SHOE SALES: "customers tend to buy shoes that match the color of their shirt." For the real-world datasets, we search for studies on these datasets on Google Scholar and compare our hypotheses with findings from the literature. We confirm the validity of some of our hypotheses and discover new insights about the tasks that previous studies did not touch upon. We show a few examples in Table 4, and the full list of hypotheses can be found in Appendix D.

Our generated hypotheses align with useful features in existing literature. For DECEPTIVE REVIEWS, we find that deceptive reviews are more likely to be emotional, use superlatives, or contain information that could not have been directly experienced. Similar findings are also found by previous studies on DECEPTIVE REVIEWS (Lai et al., 2020; Anderson and Simester, 2014; Ott et al., 2011; Li et al., 2014). For TWEET POPULARITY, we discover that tweets that are concise, with specific or relevant hashtags, or with emotional tones are more likely to be retweeted more, aligning with prior studies (Tan et al., 2014; Gligorić et al., 2019). For HEADLINE POPULARITY, we find that revealing something new or using vivid language and imagery can drive engagement from readers to click on headlines. Previous studies also find these rules apply to online news headlines (Banerjee and Urminsky, 2021; Sadoski et al., 2000).

We also discover new insights with our generated hypotheses. For the DECEPTIVE REVIEWS dataset, truthful reviews could mention the reviewer's purpose for staying at the hotel (e.g., business trip, vacation), but deceptive ones tend not to have this information. For HEADLINE POPULARITY, we find that headlines that frame the content in a personal or relatable way are clicked more. For TWEET POPULARITY, tweets that mention influential individuals or organizations are more likely to be retweeted.

Intriguingly, one of our hypotheses contradicts a feature engineering result. Ott et al. (2011) find that the token "future" is associated with deceptive reviews, while one of our hypotheses says that mentions of "past experiences or future travel plans" are indicative of truthfulness. This discrepancy is interesting, because the context for the token "future" is unclear. It could be in the context of future plans but could also be as a complaint about "never going to stay at the hotel in the future." Feature engineering is limited by contextual ambiguity, whereas our generated hypotheses and their interpretation by LLMs overcome such limitations.

Dataset	Finding	Supported/Novel
DECEPTIVE REVIEWS	Deceptive reviews contain more emotional terms.	Li et al. (2014)
	Truthful reviews would mention weddings or special occasions.	
HEADLINE POPULARITY	Using vivid language and imagery helps.	Banerjee and Urminsky (2021)
	Headlines that frame the content in a personal or relatable way are clicked more.	
TWEET POPULARITY	Tweets with emotional tones are retweeted more.	Tan et al. (2014)
	Mentioning influential individuals or organizations leads to more retweets.	

Table 4: Selected examples of generated hypotheses (on the real-world datasets) and whether they support existing findings or are novel.

Our automatic evaluation of hypothesis quality also reflects negative findings. Given mixed evidence from previous literature on the effect of "reading ease" on headline clicks, Banerjee and Urminsky (2021) finds that reading ease negatively impacts click-through rates in HEADLINE POPULARITY through careful feature engineering. Consistent with this result, we found that the hypotheses that claim "straightforward" and "clear" writing to be indicative of higher click-through rates have relatively lower accuracies during training.

5 Additional Related Work

Concept/pattern discovery. Our work is connected to many recent studies on using LLMs to propose "hypotheses", notably, Qiu et al. (2024) and Zhong et al. (2023). Qiu et al. (2024) is motivated by testing the ability of LLMs to perform human-like induction reasoning, and Zhong et al. (2023) aims to support open-ended exploration. Similar to Qiu et al. (2024), Tenenbaum et al. (2011) is motivated by human inductive reasoning and examines concept induction in synthetic settings. Ellis et al. (2020) further learns to program concepts. Yang et al. (2024a) performs LLM-based inductive reasoning with a dataset that requires existing fact-rule pairs, which is not applicable in our real-world problems. Romera-Paredes et al. (2024) generates programs that lead to mathematical discovery. Similar to Zhong et al. (2023), Pham et al. (2024) generates and refine a list of topics to achieve interpretable topic modeling for open-ended exploration. Honovich et al. (2022) explores the deduction of task description from examples. Additionally, Qi et al. (2023), Wang et al. (2024), and Baek et al. (2024) use LLMs to generate hypotheses from previous literature. Yang et al. (2024b) tries to generate hypotheses from raw web corpus, but their method is not automated or scalable as it requires human annotated hypotheses from existing literature. Our work, in contrast, focuses on hypothesis generation between the input and the label for real-world challenging tasks and uses a UCB-style reward to propose novel algorithms.

Reasoning with LLMs. Although it is not our primary goal, our results show that hypothesis-based classifiers can outperform few-shot prompting. As hypotheses may be viewed as a form of reasoning, it is related to reasoning with LLMs (Wei et al., 2022; Wang et al., 2023, *i.a.*). In particular, our work differs from chain-of-thought reasoning because no predefined reasoning structure is available. Moreover, an important distinction between reasoning and hypothesis generation is that the former leverages established reasoning, while the latter requires both proposition and verification of the hypotheses, to discover unknown knowledge.

LLMs for (social) sciences. Increasing attention has been brought to the use of LLMs in social science research (Ziems et al., 2024; Kim and Lee, 2023, *i.a.*). Our experiments demonstrate the potential of LLMs in generating hypotheses for social science research to discover unknown knowledge in the data. Furthermore, our approach can be extended to natural sciences for general scientific discovery.

6 Conclusion & Further Discussion

In this work, we propose **HypoGeniC**, a novel data-driven and automated method that leverages LLMs to generate hypotheses with the goal of discovering unknown knowledge. With **HypoGeniC**, we are not only are able to generate human-interpretable hypotheses but also achieve better predictive performance against competitive baselines and even oracles. Furthermore, our method can generalize well with different models and datasets, including open models. Notably, with our generated hypotheses, we uncover new insights in real-world tasks that are widely studied in social sciences.

The key to success in **HypoGeniC** is not that LLMs remembers the correct hypotheses, but lies in their ability to "hallucinate" and combine potentially relevant concepts. The exploration-exploitation process then identifies the valuable hypotheses. **HypoGeniC** can be directly applied to complex social science tasks. We encourage future work to explore hypothesis generation that requires additional modalities and/or leverages existing literature along with past observations.

7 Limitations

We address common concerns using a Q&A format.

Q: Why only experiment with social science tasks?

A: Math and physics problems and hypotheses are hard to represent in natural language and usually require symbolic parsers (Trinh et al., 2024). We leverage LLMs to perform tasks that it is naturally adept at, which lead us to social science tasks. We find that **HypoGeniC** demonstrates strong results for the selected tasks, indicating new possibilities in using LLMs for scientific discovery. We leave extending our framework to natural science tasks as future work.

Q: Why is **HypoGeniC** effective, given that the accuracy improvement is not significant in some settings?

A: Even if there is no significant improvement in accuracy, the benefits of **HypoGeniC** are found in the quality of hypotheses. We find that the generated hypotheses discover new patterns that were previously unseen, as discussed in § 4.3. Additionally, it is worth noting that LLMs are imperfect at reasoning. Thus, hypothesis-based inference with LLMs may not accurately reflect the quality of the hypotheses.

Q: Since you worked on some old datasets, what if the LLMs have pre-trained knowledge about these tasks?

A: In Table 1, the zero/few-shot learning results suggest that the models cannot solve the tasks by memorizing the data. Additionally in § 4.3, we show that **HypoGeniC** reveal new hypotheses, based on the literature space that we can manually search. Even if the models have been pre-trained on the datasets, these hypotheses were not reported in previous literature. This suggests that even experienced researchers still struggle in finding the hypotheses that **HypoGeniC** generate.

Q: What hyperparameters have you tried?

A: We aim to provide a robust framework for hypothesis generation, as opposed to focusing on the optimization of results. Thus, we did not perform an extensive hyperparameter search with the generation portion of **HypoGeniC**. We did not adjust the value of k, which determines $\mathcal{H}_{\text{top}}$ in Algorithm 1 to maintain efficiency. Additionally, we only considered the effect of using a hypothesis bank size of 3 and 20 to only test using an extremely small hypothesis bank size and a large one. The ideal hypothesis bank size may require further investigation. Finally, we only tested the size of our wrong example bank w_{max} as 10 to strike a balance between context window sizes and generation of good quality hypotheses. We believe that a more thorough hyperparameter search could improve the performance of our methodology.

Q: How costly is your approach?

A: HypoGeniC has high latency, specifically when using inference methods that require multiple prompts. For example, the filter and weighted vote inference policy requires iterating through the top hypotheses to determine relevance and then performing inference if it is relevant. For single-step adaptive inference and best accuracy hypothesis, however, **HypoGeniC** is efficient. Given that we request reasoning for all inference prompts, the procedure can be time-consuming and require financial costs (e.g., GPT-3.5-turbo takes $2.05 on average over 76 experiments with an average of 1.5 hours per experiment). This concern is alleviated when using open models. However, all these processes are still relatively cheap compared to human efforts.

Q: What are some potential risks of hypothesis generation?

A: One potential risk of hypothesis generation is that there is little guard regarding steorotypes and biases being confirmed if given data that may seem to enforce them. As a result, it can be potentially harmful to use **HypoGeniC** in a real-world setting without proper oversight. Additionally, if the data reveals personal information regarding people, there is no guarantee that the hypotheses generated will not reveal this information. We highly recommend human-AI collaboration in using **HypoGeniC** to ensure that the generated hypotheses are ethical and unbiased.

Acknowledgments

We thank the anonymous reviewers for their suggestions. We also thank members of the Chicago Human+AI Lab for their helpful comments. This work is supported in part by NSF grants IIS-2126602.

References

Eric T Anderson and Duncan I Simester. 2014. Reviews without a purchase: Low ratings, loyal customers, and deception. *Journal of Marketing Research*, 51(3):249–269.

Anthropic. 2023. Claude 2.

Peter Auer. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422.

Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. 2024. ResearchAgent: Iterative research idea generation over scientific literature with large language models. *Preprint*, arXiv:2404.07738.

Akshina Banerjee and Oleg Urminsky. 2021. The language that drives engagement: A systematic large-scale analysis of headline experiments. *Social Science Research Network*.

Tom 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 Ziegler, Jeffrey Wu, Clemens Winter, Chris 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. In *Proceedings of NeurIPS*, volume 33, pages 1877–1901.

Kevin Ellis, Catherine Wong, Maxwell Nye, Mathias Sablé-Meyer, Luc Cary, Lucas Morales, Luke Hewitt, Armando Solar-Lezama, and Joshua B. Tenenbaum. 2020. DreamCoder: growing generalizable, interpretable knowledge with wake-sleep Bayesian program learning. *Philosophical Transactions of the Royal Society A*, 381.

Kristina Gligorić, Ashton Anderson, and Robert West. 2019. Causal effects of brevity on style and success in social media. In *Proceedings of ACM HCI*.

Pär Anders Granhag and Aldert Vrij. 2005. Deception detection. *Psychology and law: An empirical perspective*, pages 43–92.

Or Honovich, Uri Shaham, Samuel R. Bowman, and Omer Levy. 2022. Instruction induction: From few examples to natural language task descriptions. In *Proceedings of ACL*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*, abs/2001.08361.

Junsol Kim and Byungkyu Lee. 2023. AI-augmented surveys: Leveraging large language models and surveys for opinion prediction. *Preprint*, arXiv:2305.09620.

Vivian Lai, Han Liu, and Chenhao Tan. 2020. "Why is 'Chicago' deceptive?" Towards building model-driven tutorials for humans. In *Proceedings of CHI*.

Vivian Lai and Chenhao Tan. 2019. On human predictions with explanations and predictions of machine learning models: A case study on deception detection. In *Proceedings of FAccT*.

Jiwei Li, Myle Ott, and Claire Cardie. 2013. Identifying manipulated offerings on review portals. In *Proceedings of EMNLP*.

Jiwei Li, Myle Ott, Claire Cardie, and Eduard Hovy. 2014. Towards a general rule for identifying deceptive opinion spam. In *Proceedings of ACL*, pages 1566–1576, Baltimore, Maryland. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *ArXiv*.

Jens Ludwig and Sendhil Mullainathan. 2024. Machine learning as a tool for hypothesis generation*. *The Quarterly Journal of Economics*, page qjad055.

Jorge Nathan Matias, Kevin Munger, Marianne Aubin Le Quere, and Charles R. Ebersole. 2021. The upworthy research archive, a time series of 32,487 experiments in U.S. media. *Scientific Data*, 8.

Mistral. 2023. Mixtral of experts.

OpenAI. 2023a. Chatgpt.

OpenAI. 2023b. Gpt-4 technical report.

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of ACL*.

Chau Minh Pham, Alexander Hoyle, Simeng Sun, and Mohit Iyyer. 2024. Topicgpt: A prompt-based topic modeling framework. In *Proceedings of NAACL*.

Biqing Qi, Kaiyan Zhang, Haoxiang Li, Kai Tian, Sihang Zeng, Zhang-Ren Chen, and Bowen Zhou. 2023. Large language models are zero shot hypothesis proposers. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Linlu Qiu, Liwei Jiang, Ximing Lu, Melanie Sclar, Valentina Pyatkin, Chandra Bhagavatula, Bailin Wang, Yoon Kim, Yejin Choi, Nouha Dziri, and Xiang Ren. 2024. Phenomenal yet puzzling: Testing inductive reasoning capabilities of language models with hypothesis refinement. In *Proceedings of ICLR*.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. 2024. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475.

Albert Rothenberg. 1995. Creative cognitive processes in kekule's discovery of the structure of the benzene molecule. *The American journal of psychology*, pages 419–438.

Mark Sadoski, Ernest T Goetz, and Maximo Rodriguez. 2000. Engaging texts: Effects of concreteness on comprehensibility, interest, and recall in four text types. *Journal of Educational Psychology*, 92(1):85.

Matthew J Salganik, Peter Sheridan Dodds, and Duncan J Watts. 2006. Experimental study of inequality and unpredictability in an artificial cultural market. *Science*, 311(5762):854–856.

Chenhao Tan, Lillian Lee, and Bo Pang. 2014. The effect of wording on message propagation: Topic- and author-controlled natural experiments on twitter. In *Proceedings of ACL*.

Joshua B. Tenenbaum, Charles Kemp, Thomas L. Griffiths, and Noah D. Goodman. 2011. How to grow a mind: Statistics, structure, and abstraction. *Science*, 331:1279 – 1285.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami,

Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*.

Trieu Trinh, Yuhuai Tony Wu, Quoc Le, He He, and Thang Luong. 2024. Solving olympiad geometry without human demonstrations. *Nature*, 625:476–482.

Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. 2024. SciMON: Scientific inspiration machines optimized for novelty. In *Proceedings of ACL*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *Proceedings of ICLR*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Proceedings of NeurIPS*.

Zonglin Yang, Li Dong, Xinya Du, Hao Cheng, Erik Cambria, Xiaodong Liu, Jianfeng Gao, and Furu Wei. 2024a. Language models as inductive reasoners. In *Proceedings of EACL*.

Zonglin Yang, Xinya Du, Junxian Li, Jie Zheng, Soujanya Poria, and Erik Cambria. 2024b. Large language models for automated open-domain scientific hypotheses discovery. In *Proceedings of ACL*.

Ruiqi Zhong, Peter Zhang, Steve Li, Jinwoo Ahn, Dan Klein, and Jacob Steinhardt. 2023. Goal driven discovery of distributional differences via language descriptions. In *Proceedings of NeurIPS*.

Caleb Ziems, William Held, Omar Shaikh, Jiaao Chen, Zhehao Zhang, and Diyi Yang. 2024. Can large language models transform computational social science? *Computational Linguistics*, pages 1–55.

A Prompts

We follow the general prompt engineering guide from Claude (Anthropic, 2023) to craft the prompts. Specifically for all the prompts we use for LLMs, we split them into instruction and user prompts. In the instruction prompt, we first set a tone and context, followed by an explicit task description, and then specify the answer format. The user prompt then includes useful information such as past examples and learned hypothesis. By the end of the user prompt, we ask the LLM to make a prediction. At generation time, we input the instruction prompt to LLMs as system prompt, wrapped by the corresponding system prompt tokens for each model. Below are some example templates for the prompts associated with each task.

A.1 Shoe Sales

```
Instruction Prompt
You're a helpful assistant. Your task is given as
 follows:
Given a set of observations, we want to generate
hypotheses that are useful for predicting the
color of the shoes given the appearance of the
person.
Please be concise and keep the hypotheses to be
one-sentence long.
Please generate them in the format of
{1. [hypothesis].
2. [hypothesis].
...
<num_hypotheses>. [hypothesis].}
Only propose <num_hypotheses> possible hypotheses
 in total.
No need to explain the hypotheses.

User Prompt
We made some observations:
··· more examples here ···
Based on the above observations, generate
<num_hypotheses> hypotheses.
Please be concise and keep the hypotheses to be
one-sentence long.
Please generate them in the format of
{1. [hypothesis].
2. [hypothesis].
...
<num_hypotheses>. [hypothesis].}
Only propose <num_hypotheses> possible hypotheses
 in total.
```

Example 1: Hypothesis Generation.

```
Instruction Prompt
You are a shoe salesman and want to recommend
shoes to customers. There are white, red, orange,
 green, blue, and black shoes.
From past experiences, you learned some patterns.
Now, at each time, you should apply the learned
pattern, given below, to a new customer and
recommend a shoe color.
Give an answer for the shoe color recommendation.
 The answer should be one color word. It has to
be one of white, red, orange, green, blue, and
black.

User Prompt
Our learned pattern: <hypothesis_high_reward>
New customer: <appearance> is buying a pair of
shoes, the shoes should be which color?
Answer:
```

Example 2: Hypothesis-based Inference.

```
Instruction Prompt
You are a shoe salesman and want to recommend
shoes to customers. There are white, red, orange,
 green, blue, and black shoes.
Give your answer for the shoe color
recommendation. The answer should be one color
word. It has to be one of white, red, orange,
green, blue, and black. If you do not have enough
 information to make a recommendation, you should
 give the answer "unknown".
Give your final answer in the format of "Final
answer: [answer]."

User Prompt
Here are some examples of customers with certain
features buying certain products:
··· more examples here ···
New customer: <appearance> is buying a pair of
shoes, the shoes should be which color?
Answer:
```

Example 3: Zero/Few-shot Inference.

```
Instruction Prompt
You are a shoe salesman and want to recommend
shoes to customers. There are white, red, orange,
 green, blue, and black shoes.
From past experiences, you learned some patterns.
For each pattern, you will also see a couple of
examples that worked for each pattern.
Choose a pattern. To do this, look at the
examples of each pattern, and see which of the
examples the current customer is closest to.
Choose the pattern corresponding to that example.
Give an answer for the shoe color recommendation.
The answer should be one word. It has to be one
of white, red, orange, green, blue, and black.
Give your final answer in the following format:
Reasoning for choosing pattern: reason,
Chosen pattern: pattern,
Reasoning for choice of prediction: reason,
Final Answer: answer

User Prompt
Here are some previously generated patterns with
some example where it predicted correcly what
color of shoe the customer bought.
<adaptive_info_prompt>
New customer: <appearance> is buying a pair of
shoes, the shoes should be which color?
Answer:
```

Example 4: Example-based Hypothesis Selection and Inference. <adaptive_info_prompt> consists of several hypotheses and the corresponding examples they got correct during generation time.

A.2 Deceptive Reviews

```
Instruction Prompt
You're a professional hotel review analyst.
Given a set of hotel reviews, we want to generate
 hypotheses that are useful for predicting
whether a review is truthful or deceptive. In
other words, we want to know whether the review
is written by a someone who actually lived in the
 hotel.
Using the given examples, please propose
<num_hypotheses> possible hypothesis pairs.
These hypotheses should identify specific
patterns that occur across the provided reviews.
Each hypothesis should contain a pair of the
following:
1. A hypothesis about what makes reviews more
likely to be truthful
2. The opposite hypothesis about what makes
reviews more likely to be deceptive
Generate them in the format of 1. [hypothesis], 2.
 [hypothesis], ... <num_hypotheses>. [hypothesis].

The hypotheses should analyze what kind of
reviews are likely to be truthful or deceptive.

User Prompt
```

```
We have seen some hotel reviews:
··· more examples here ···
Please generate hypotheses that are useful for
predicting whether a review is truthful or
deceptive.
Propose <num_hypotheses> possible hypotheses.
Generate them in the format of 1. [hypothesis], 2.
 [hypothesis], ... <num_hypotheses>. [hypothesis].

Proposed hypotheses:
```

Example 5: Hypothesis Generation.

```
Instruction Prompt
You are a professional deceptive detection agent
and your job is to determine whether a hotel
review is truthful or deceptive.
In other words, we want to know whether the
review is written by someone who had real
experiences with the hotel.
From past experiences, you learned a pattern.
You need to determine whether each of the
patterns holds for the current hotel review, and
also predict whether the current hotel review is
truthful or deceptive.
Give an answer. The answer should be one word (
truthful or deceptive).
Give your final answer in the format of {Final
answer: answer}

User Prompt
Our learned pattern: <hypothesis_high_reward>
A hotel review is the following: <review>
Given the pattern you learned above, give an
answer of whether the hotel review above is
deceptive or truthful.
Think step by step.
First step: Think about which pattern can be
applied to the hotel review.
Second step: Based on the pattern, is this hotel
review deceptive or truthful?
```

Example 6: Hypothesis-based Inference.

```
Instruction Prompt
You are a deceptive detection agent and want to
determine whether a hotel review is truthful or
deceptive.
In other words, we want to know whether the
review is written by a someone who actually lived
 in the hotel.
You need to determine whether this pattern holds
for the current hotel review, and also predict
whether the current hotel review is truthful or
deceptive.
Give an answer. The answer should be one word (
truthful or deceptive).

User Prompt
We have seen some hotel reviews:
··· more examples here ···
A hotel review is the following: <review>
Is this hotel review truthful or deceptive?
Answer:
```

Example 7: Zero/Few-shot Inference.

```
Instruction Prompt
You are a professional hotel review analyst and
you are able to determine whether a hotel review
is deceptive or truthful.
In other words, your job is to analyze if a hotel
 review review is written by someone who had
genuine experiences with the hotel.
From past experiences, you learned some patterns.
For each pattern, you will also see a couple of
examples that worked for each pattern.
First step: take a careful look at the examples
associated with each pattern, and see which set
of examples the current hotel review is most
similar with. Choose and repeat the pattern
corresponding to that examples set.
Next, apply the pattern on the new sample to
determine whether the new hotel review is
deceptive or truthful.
Finally, give an answer. The answer should be one
 word (deceptive or truthful).
Please give your final answer in the following
format:
Reasoning for choosing pattern: reason,
Chosen pattern: pattern,
Reasoning for choice of prediction: reason,
Final Answer: answer

User Prompt
Here are some previously generated patterns with
some example where it predicted correctly if a
hotel review is deceptive or truthful.
<adaptive_info_prompt>
A hotel review is the following: <review>
Is this hotel review truthful or deceptive?
Think step-by-step.
Step 1: Look at the new hotel review and compare
it with the set of examples associated with each
provided pattern.
Step 2: Find the set of examples that is the most
 similar to the new hotel review, pick and repeat
 the pattern associated with that set of examples.

Step 3: Apply the pattern you picked to the new
hotel review and predict whether the new hotel
review is deceptive or truthful.
Step 4: Give your final answer.
Answer:
```

Example 8: Example-based Hypothesis Selection and Inference. <adaptive_info_prompt> consists of several hypotheses and the corresponding examples they got correct during generation time.

A.3 Headlines With More Clicks

```
Instruction Prompt
You are a professional writer for an online
newspaper company.
Given a pair of headlines created for the same
article, you are asked to determine which will
get more clicks. It is likely that the pair of
headlines shares similarities, so please focus on
 their differences.
What difference in two headlines leads to more
clicks on one than the other?
You will be given a set of observations of the
format:
Headline 1: [headline]
Headline 2: [headline]
Observation: [observation].
Based on the observations, please generate
hypotheses that are useful for explaining why one
 headline out of the pair gets more clicked than
the other.
These hypotheses should identify patterns,
phrases, wordings etc. that occur across the
provided examples. They should also be
generalizable to new instances.
Please propose <num_hypotheses> possible
hypotheses and generate them in the format of 1.
[hypothesis], 2. [hypothesis], ...
<num_hypotheses>. [hypothesis].

User Prompt
Here are the observations:
··· more examples here ···
Please generate hypotheses that can help
determine which headlines have more clicks.
Please propose <num_hypotheses> possible
hypotheses.
Generate them in the format of 1. [hypothesis], 2.
 [hypothesis], ... <num_hypotheses>. [hypothesis].

Proposed hypotheses:
```

Example 9: Hypothesis Generation.

```
Instruction Prompt
You are a professional writer for an online
newspaper company.
```

```
Given a pair of headlines created for the same
article, you are asked to determine which will
get more clicks. It is likely that the pair of
headlines shares similarities, so please focus on
 their differences.
From past experiences, you learned some patterns.
Now, at each time, you should apply the learned
pattern to a new pair of headlines that are
created for a new article and determine which
headline gets clicked more.
The answer for the higher clicks should be in the
 form "Headline _" where _ is either 1 or 2.
Please give your final answer in the format of {
Final Answer: Headline _.}

User Prompt
Learned pattern: <hypothesis_high_reward>
Given the pattern you learned above, predict
which of the following headlines will get more
clicks:
Headline 1: <headline_1>
Headline 2: <headline_2>
Think step by step.
Step 1: Think about whether the pattern can be
applied to the headlines.
Step 2: Analyze the difference between "Headline
1" and "Headline 2".
Step 3: Based on the pattern, which headline is
likely to get more clicks?
```

Example 10: Hypothesis-based Inference.

```
Instruction Prompt
YYou are a writer for an online newspaper company.
 So you are excellent at determining which
headlines are more likely to cause users to click
 on the article.
You will be given two headlines, and determine
which headline was clicked more often.
You are only to give your answer.
The answer for the higher clicks should be of the
 form "Headline _" where _ is either 1 or 2.
Give your final answer in the following format:
"Answer: Headline _"

User Prompt
Here are some previous examples to help you:
··· more examples here ···
Which of the following headlines has more clicks:
Headline 1: <headline_1>
Headline 2: <headline_2>
```

Example 11: Zero/Few-shot Inference.

```
Instruction Prompt
You are a professional writer for an online
newspaper company.
You are excellent at determining which headlines
are more likely to be clicked by users.
From past experiences, you learned some patterns.
For each pattern, you will also see a couple of
examples that worked for each pattern.
Please choose a pattern. To do this, look at the
examples associated with each pattern, and find
which set of the examples are closest to the
given pair of headlines.
Please choose the pattern corresponding to that
set of examples.
The answer for the higher clicks should be of the
 form "Headline _" where _ is either 1 or 2.
Please give your final answer in the following
format:
Reasoning for choosing pattern: reason,
Chosen pattern: pattern,
Reasoning for choice of prediction: reason,
Final Answer: answer

User Prompt
Here are some previously generated patterns with
some examples where it predicted which one of the
 pair of headlines got more clicks.
<adaptive_info_prompt>
Which one out of the following pair of headlines
will get more clicks?
Headline 1: <headline_1>
Headline 2: <headline_2>
Think step by step.
Step 1: Look at the new pair of headlines and
compare them with the examples associated with
each pattern.
Step 2: Find the set of examples that is closest
to the given pair of headlines, and pick the
pattern associated with that set of examples.
Step 3: Apply the picked pattern to the new pair
of headlines. Based on that pattern, think about
which one out of the pair of headlines will get
more clicks.
Step 4: Give your final answer.
```

Example 12: Example-based Hypothesis Selection and Inference. <adaptive_info_prompt> consists of several hypotheses and the corresponding examples they got correct during generation time.

A.4 Retweeted More

```
Instruction Prompt
You are a social media expert. You are an expert
at determining which tweet will be retweeted more.

Given a set of observations, you want to
generation hypotheses that will help predict
which tweet out of a pair of tweets is more
likely to be retweeted.
Please note that the paired tweets are about the
same content and are posted by the same user, so
you should focus on the wording difference
between the two tweets in each pair.
Please propose <num_hypotheses> possible
hypotheses.
Please generate them in the format of 1. [
hypothesis], 2. [hypothesis], ...
<num_hypotheses>. [hypothesis].
Please make the hypotheses general enough to be
applicable to new observations.

User Prompt
We made some observations:
··· more examples here ···
Generate hypotheses that are useful for
predicting which tweet out of a pair of tweets is
 more likely to be retweeted.
Please note that the paired tweets are about the
same content and are posted by the same user, so
you should focus on the wording difference
between the two tweets in each pair.
Please propose <num_hypotheses> possible
hypotheses.
Please generate them in the format of 1. [
hypothesis], 2. [hypothesis], ...
<num_hypotheses>. [hypothesis].
Proposed hypotheses:
```

Example 13: Hypothesis Generation.

```
Instruction Prompt
You are a social media expert.
Given a pair of tweets, you are asked to predict
which tweet will be retweeted more.
Please note that the paired tweets are about the
same content and are posted by the same user, so
you should focus on the wording difference
between the two tweets.
From past experiences, you learned a pattern.
Now, at each time, you should apply a learned
pattern to a pair of tweets and determine which
one will get more retweets.
The answer for the higher retweets should be of
the form "the _ tweet" where _ is either first or
 second.
Please give your final answer in the format of {
Final answer: the _ tweet}

User Prompt
Our learned pattern: <hypothesis_high_reward>
The first tweet: <first_tweet>
The second tweet: <second_tweet>
Given the pattern you learned above, predict
which one of the two tweets will get more
```

```
retweets.
Think step by step.
First step: Think about if the pattern can be
applied to the tweets.
Second step: Analyze the textual difference
between the two tweets.
Third step: Based on the pattern, which tweet is
more likely to get more retweets?
Final step: Give your final answer in the format
of {Final answer: the _ tweet}
Final answer:
```

Example 14: Hypothesis-based Inference.

```
Instruction Prompt
You are a social media expert.
Given a pair of tweets, you are asked to predict
which tweet will be retweeted more.
Please note that the paired tweets are about the
same content and are posted by the same user, so
you should focus on the wording difference
between the two tweets.
The answer for the higher retweets should be of
the form "the _ tweet" where _ is either first or
 second.
Please give your final answer in the format of {
Final answer: the _ tweet}

User Prompt
Here are some examples:
··· more examples here ···
The first tweet: <first_tweet>
The second tweet: <second_tweet>
Which one of the two tweets will get more
retweets?
```

Example 15: Zero/Few-shot Inference.

```
Instruction Prompt
You are a social media expert.
Given a pair of tweets, you are asked to predict
which tweet will be retweeted more.
Please note that the paired tweets are about the
same content and are posted by the same user, so
you should focus on the wording difference
between the two tweets.
From past experiences, you learned some patterns.
You should apply a learned pattern to a pair of
tweets and determine which one will get more
retweets.
For each pattern, you will also see a couple of
examples that worked for each pattern.
Please choose a pattern. To do this, look at the
examples associated with each pattern, and find
which set of the examples are closest to the
given pair of tweets.
Please choose the pattern corresponding to that
set of examples.
Please give your final answer in the following
format:
Reasoning for choosing pattern: reason,
Chosen pattern: pattern,
Reasoning for choice of prediction: reason,
Final Answer: answer

User Prompt
Here are some previously generated patterns with
some examples where it predicted which tweet will
 will be retweeted more.
<adaptive_info_prompt>
The first tweet: <first_tweet>
The second tweet: <second_tweet>
Which one of the two tweets will get more
retweets?
Think step by step.
Step 1: Look at the new pair of tweets and
compare them with the examples associated with
each pattern.
Step 2: Find the set of examples that is closest
to the given pair of tweets, and pick the pattern
 associated with that set of examples.
Step 3: Analyze the textual difference between
the two tweets.
Step 4: Apply the picked pattern to the new pair
of tweets. Based on that pattern, think about
which one out of the pair of headlines will get
more clicks.
Step 5: Give your final answer.
```

Example 16: Example-based Hypothesis Selection and Inference. <adaptive_info_prompt> consists of several hypotheses and the corresponding examples they got correct during generation time.

B Implementation and Setup Details

B.1 HypoGeniC implementation

Sampling When initializing the rewards of newly generated hypotheses, we use the examples in the wrong example bank to do so. Given that we work in a low data regime, for hypotheses generated near the end of the training loop, the accuracies of hypotheses are likely to be biased. To counter this phenomenon, we also allow for the hypotheses to use the initial examples $\mathcal{S}_{\text{init}}$ for initializing rewards. By allowing the hypotheses to initialize reward with more examples, the accuracy lies closer to its true value, allowing for fair comparison between earlier generated hypotheses and newer ones.

Dynamic hypotheses update In Algorithm 1, we display how we generate and update the hypotheses pool $\mathcal{H}$. In particular, we add an example s to the wrong example bank $\mathcal{W}$ if the number of hypotheses that incorrectly predict s is greater than w_{hyp}. In our implementation, we use a linearly increasing w_{hyp} as training time t increases. This allows our algorithm to update the hypotheses more frequently at early stage of training, and less frequently at the end.

B.2 Inference method implementations

Filter and weighted vote In order to filter the hypotheses, we iterate through the top k hypotheses ranked by reward. For each hypothesis, we ask the Large Language Model (LLM) if it is relevant. Thereafter, for each of the relevant hypotheses, the LLM is prompted to use the hypothesis to make predictions. Then, for each predicted label, we add up the accuracy scores from the hypotheses that outputted that particular label. The final label is the one that has highest total accuracy score.

One-step adaptive and two-step adaptive inference The detailed framework of our adaptive inference methods is split into two parts - hypotheses pruning and hypotheses selection. In the case where we have a large number of hypotheses, it is likely that some hypotheses in $\mathcal{H}$ have overlaps or are paraphrases of each other.

We address this issue with the following procedure:

1. During training, we record the examples that each hypothesis correctly predicts.

2. Then we create one-hot encodings for each hypothesis, where the i-th element of the one-hot encoding is 1 if the hypothesis correctly predicts the i-th example, and 0 otherwise. We subsequently

compute a similarity matrix between each pair of hypotheses by taking the pairwise cosine similarities.

3. Lastly, we create a linear program with the objective of maximizing the sum of accuracies of the selected hypotheses, subject to the constraint that every pair of the selected hypotheses has a similarity score below a predefined threshold γ.

After pruning the set of hypotheses, we prompt the LLM to pick one hypothesis for its final prediction, as described in § 2.2. For the single-step adaptive inference, we ask the LLM to select a hypothesis and make a prediction in one prompt. On the other hand, with the two-step adaptive inference, we first prompt the LLM to select a hypothesis and then prompt the LLM again to make a prediction based on the selected hypothesis.

B.3 Hyperparameters

For the training stage, we set a limit on the hypothesis bank size, experimenting with sizes $H = 3$ and $H = 20$ to determine the impact of utilizing a larger number of hypotheses. Throughout all the experiments, we use the reward coefficient $\alpha = 0.5$, $w_{max} = 10$, $\texttt{num\_init} = 10$, and we have two different sets of the rest of hyperparameters for hypothesis bank sizes of 3 and 20.

- With $H = 3$, we use $k = 2$ and generate 1 hypothesis per update. For inference, we employ all 3 hypotheses for filter and weighted vote. For single-step and two-step adaptive inference, we use all 3 hypotheses with $\gamma = 0.3$ and provide 5 examples to each hypothesis.
- In the case of $H = 20$, we use $k = 10$ and generate 5 hypotheses per update. Then we take the top 5 hypotheses, ranked by their training accuracies, for filter and weighted vote. For single-step and two-step adaptive inference, we use the top 5 hypotheses with $\gamma = 0.7$ and provide 5 examples each.

B.4 Licensing Details

The DECEPTIVE REVIEWS and TWEET POPULARITY datasets have not been released with any licenses, but are free to use for research purposes based upon the authors. The HEADLINE POPULARITY dataset is released under the Creative Commons Attribution 4.0 International License. The SHOE SALES dataset will be released under the same licensing as this work, CC BY 4.0 License, should it be accepted.

In regards to models, we find that GPT-3.5-turbo and Claude-2.1 are all proprietary models and are not released under any open-source licenses. On the other hand, Mixtral is released under the Apache License 2.0. RoBERTa is not released under specific licensing but is free to use for research purposes. However, Llama-2-7B is released under their own licensing found at https://ai.meta.com/llama/license/.

Per our extensive search, we find that we are in compliance with the licensing agreements of all the datasets and models used in this work.

C Detailed Results

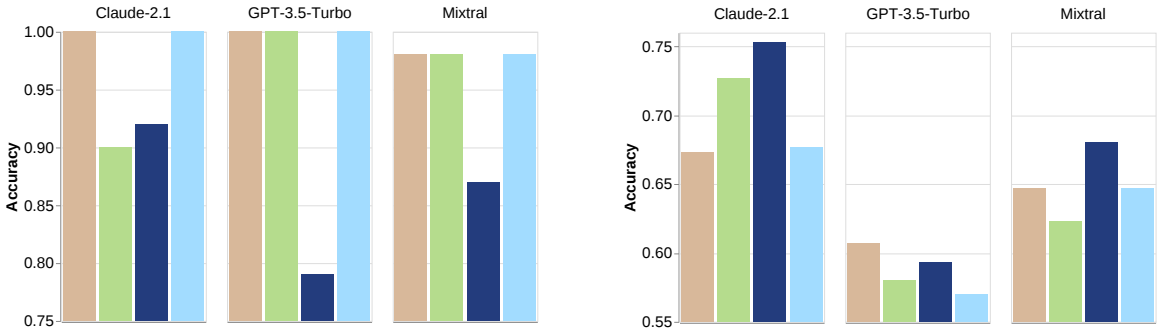
C.1 HypoGeniC Performance across inference strategies

Figure 3 presents the best results for all of our inference strategies, considering every dataset and all hyperparameter configurations.

For SHOE SALES, we observe that all the models perform effectively by using the best hypothesis inference strategy. Surprisingly, Mixtral is unable to perform perfectly. This is because despite generating the hypothesis that fully describes the data, Mixtral opts not to apply the hypotheses, favoring to choose a random label for the sake of "variety". Both GPT-3.5-turbo and Mixtral display similar patterns across the inference strategies, with best-accuracy hypothesis, filter and weighted vote, and two-step adaptive inference all having comparable performance. However, for all models we find single-step adaptive inference drops in accuracy. Given that two-step adaptive inference performs well, it is likely that the long prompt causes the model difficulty in choosing the correct hypotheses. For Claude-2.1, we see that filter and weighted vote drops in performance. As this method searches for relevant hypotheses, the model is likely finding that inaccurate patterns relevant, which end up outweighing the inference of the best hypothesis.

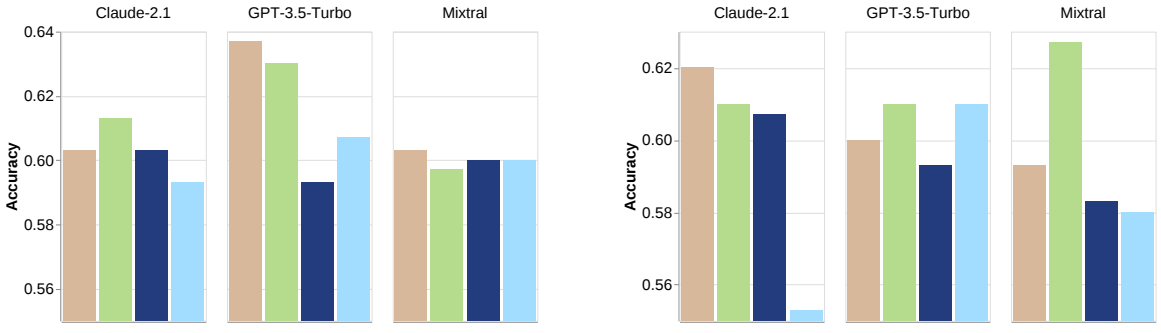
For DECEPTIVE REVIEWS, Claude-2.1 is the best performing model across all inference policies. Across the models, we highlight that single-step adaptive inference method works best for this dataset. In this inference method, the prompt specifically includes the aims of determining if a review is deceptive. This likely helps the model use the context provided to better decide which set of example resembles the test example most. Hence, splitting up the prompt may have caused performance to suffer.

We find that HEADLINE POPULARITY is the most challenging dataset. As mentioned in § 3.1, the original dataset was created with both images and headlines paired together. In our version of the dataset, we only use the headlines, so we are missing a crucial variable that contributes to understanding click behavior. Therefore, based off only headlines, it is difficult to generate hypotheses that truly capture the data. Despite this challenge, we note that our hypotheses can still adeptly capture a large portion of data with 63.7% being our highest accuracy. Specifically, we find that the best-accuracy hypothesis strategy performs best. We also note that filter and weighted vote can provide strong performance as in the case of Claude-2.1 and GPT-3.5-turbo, suggesting that hypotheses corroborating with each other can lead to better performance. We observe that GPT-3.5-turbo is the best performing model here, with all inference policies (aside from single-step adaptive) having high



(a) SHOE SALES

(b) DECEPTIVE REVIEWS



(c) HEADLINE POPULARITY

(d) TWEET POPULARITY

Figure 3: **HypoGeniC** results with different inference strategies. Best-accuracy hypothesis is sufficient for getting good performance on SHOE SALES and HEADLINE POPULARITY. Single-step adaptive hypothesis-based inference is the most effective on DECEPTIVE REVIEWS. Filter and weighted vote is best on TWEET POPULARITY.

accuracy.

Finally, over the TWEET POPULARITY dataset, we find that the filter and weighted vote is the best choice for inference policy, with it being the best inference method for GPT-3.5-turbo and Mixtral. This indicates that using hypotheses in conjunction is useful as multiple variables together adeptly characterize the dataset. The performance of the rest of the inference policies has no clear pattern over this dataset.

We also present our results with confidence intervals. We specifically see that compared to the Oracle Methods, **HypoGeniC** shows performance statistically significant benefits when comparing to the 200 training examples for HEADLINE POPULARITY and TWEET POPULARITY. However, this is not the case for DECEPTIVE REVIEWS, because there are word level features that make the task easier for unsupervised methods. We note that **HypoGeniC** has statistically significant performance increases for DECEPTIVE REVIEWS with Claude-2.1 and Mixtral and for TWEET POPULARITY with Claude-2.1 and Mixtral.

C.2 HypoGeniC Performance across training examples

Figure 4 presents the results for the performance of **HypoGeniC** with Claude-2.1 as the training examples change. We observe that for all of our datasets, **HypoGeniC** outperforms zero-shot and few-shot learning generally for all training examples in SHOE SALES and TWEET POPULARITY. In HEADLINE POPULARITY, we find that the model needs to use 200 examples to outperform them. We highlight that **HypoGeniC** outperforms the No Updates method for all training examples across the four datasets when using a hypothesis bank size of 20. When using a hypothesis bank size of 3, we find that in TWEET POPULARITY, **HypoGeniC** is able to outperform the No Updates method, but is unable to as the training examples increase. In SHOE SALES we observe that it is largely worse because we set k (as discussed in § 2.1) to be 1, which causes difficulty in finding the best hypothesis. It is unclear what the optimal number of training examples is across the datasets, as using more examples does not necessarily increase accuracy.

Figure 5 displays the accuracy for **HypoGeniC** with GPT-3.5-turbo for the different training examples. We observe that unlike **HypoGeniC** performance with Claude-2.1, our results are mixed for when our method outperforms the few shot inference. Specifically, in TWEET POPULARITY, the few shot inference surpasses our results, indicating that in this set hypotheses provide less benefits than using examples. As **HypoGeniC** exceeds the accuracy of zero shot's, the proposed method still provides benefits to the base model. Similar to the results on Claude-2.1, we outperform RoBERTa and Llama-2-7B on all datasets aside on DECEPTIVE REVIEWS for all training examples. **HypoGeniC** surpasses the performance of the No Update strategy generally for all training examples. We note that due to the limited context window of GPT-3.5-turbo, the No Update strategy fails as it is unable to accept training exam-

Models	Methods	SHOE SALES	DECEPTIVE REVIEWS	HEADLINE POPULARITY	TWEET POPULARITY
RoBERTa (Oracle)	Train 200	100.0 ± 0.0	84.0 ± 4.2	49.0 ± 5.7	50.7 ± 5.7
	Train 1000	100.0 ± 0.0	91.0 ± 3.2	60.0 ± 5.5	62.0 ± 5.5
Llama-2-7B (Oracle)	Train 200	100.0 ± 0.0	88.7 ± 3.6	49.7 ± 5.7	50.3 ± 5.7
	Train 1000	100.0 ± 0.0	92.3 ± 3.0	60.0 ± 5.5	51.3 ± 5.7
Claude-2.1	Few shot	75.0 ± 4.9	51.0 ± 5.7	60.0 ± 5.5	0.3* ± 0.6
	HypoGeniC	**100.0 ± 0.0**	**75.3 ± 4.9**	**61.3 ± 5.5**	**62.0 ± 5.5**
Mixtral	Few shot	79.0 ± 4.6	56.3 ± 5.6	55.3 ± 5.6	48.7 ± 5.7
	HypoGeniC	**98.0 ± 1.6**	**68.0 ± 5.3**	**60.3 ± 5.5**	**62.7 ± 5.5**
GPT-3.5-turbo	Few shot	49.0 ± 5.7	55.0 ± 5.6	60.0 ± 5.5	**62.0± 5.5**
	HypoGeniC	**100.0 ± 0.0**	**60.7 ± 5.5**	**63.7 ± 5.4**	61.0 ± 5.5

Table 5: Table with 95% confidence interval for Few shot results and **HypoGeniC** for our best results.

ples. **HypoGeniC** effectively bypasses this issue by iteratively going through test examples, as opposed to feeding them into the model all at once.

In, Figure 6, the performance of **HypoGeniC** for varying training examples with Mixtral is shown. **HypoGeniC** outperforms the zero shot and few shot strategies for all datasets, aside from SHOE SALES, where the proposed method requires 200 examples to outperform few shot learning. Similary, we note that **HypoGeniC** surpasses the performance of RoBERTa and Llama-2-7B for HEADLINE POPULARITY, TWEET POPULARITY, and generally for SHOE SALES. As mentioned in Appendix C.1, despite Mixtral finding the best hypothesis, it occasionally refuses to choose the correct label to encourage "variety", which causes RoBERTa and Llama-2-7B to outpeform **HypoGeniC**. In comparison to the No Update results, we find that in DECEPTIVE REVIEWS and HEADLINE POPULARITY, **HypoGeniC** matches or exceeds this method. For SHOE SALES, we find that with hypothesis bank 3, **HypoGeniC** must use 200 examples, to finally converge to the correct hypothesis. On the other hand, for TWEET POPULARITY, No Update surpasses the **HypoGeniC** with hypothesis bank size 3 after using 200 training examples. This may occur as using 3 hypotheses is too limited to adeptly describe the dataset, causing accuracy to suffer.

C.3 Full OOD results

Table 6 shows results for the OOD deceptive reviews dataset for all inference strategies for each model.

We find that **HypoGeniC** outperforms both zero shot and few shot learning across all models and inference policies. The best-accuracy hypothesis and two-step adaptive inference methods are the most robust, showing an average increase of 3.7% and 3.6% respectively. We claim that although the filter and weighted vote strategy at first glance may seem to have mixed performance, the method is still robust. The drop in accuracy for Mixtral with filter and weighted is minimal (1%), and both GPT-3.5-turbo and Claude-2.1 exhibit increases in accuracy. Hence, the inference policy is consistent across DECEPTIVE REVIEWS and the OOD deceptive review datset. Interestingly, the single-step adaptive inference method exhibits drops in performance despite being the best performing inference model in DECEPTIVE REVIEWS. In single-step adaptive inference, the LLM sees both the hypotheses with the sets of examples along with the final question of determining whether the review is deceptive. Even though the LLM is prompted to only use one chosen hypotheses, these training examples from DECEPTIVE REVIEWS negatively impact the model because they are part of the context and are thus inherently used by LLMs. On the other hand, for two-step adaptive inference, since there is a dedicated prompt for hypothesis selection, the application of the hypothesis is unaffected from the DECEPTIVE REVIEWS training samples.

D Qualitative Analysis on Generated Hypotheses

We include findings from the generated hypotheses on DECEPTIVE REVIEWS, HEADLINE POPULARITY, and TWEET POPULARITY datasets in Table 7. The table shows that the a good number of the hypotheses are supported by existing findings, while others are novel. This suggests that the generated hypotheses are grounded in existing literature and can be used to guide future research.

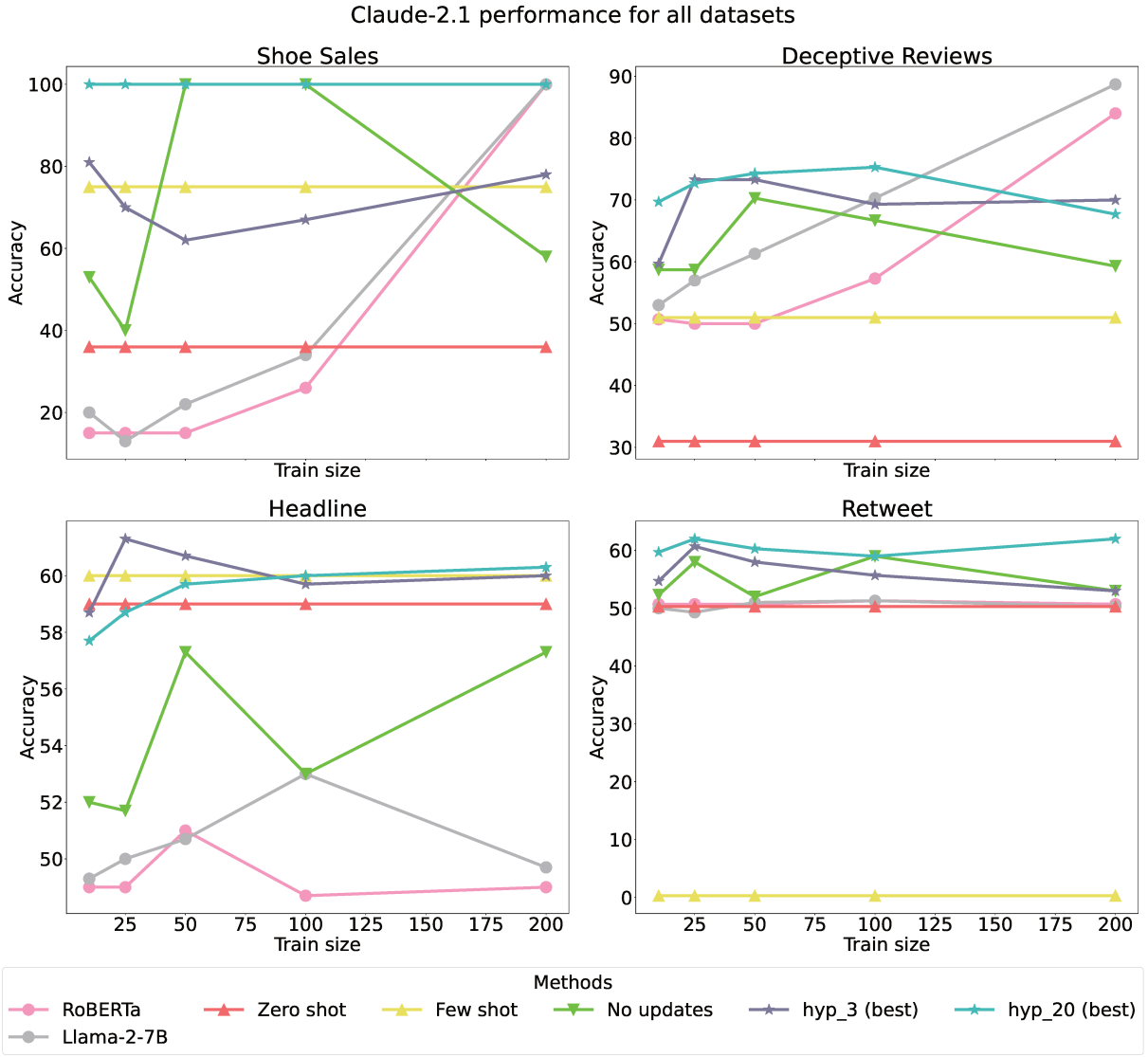


Figure 4: Claude-2.1 results for baselines, **HypoGeniC** (no update), and **HypoGeniC** (best) with hypothesis bank size 3 and 20 across multiple training samples

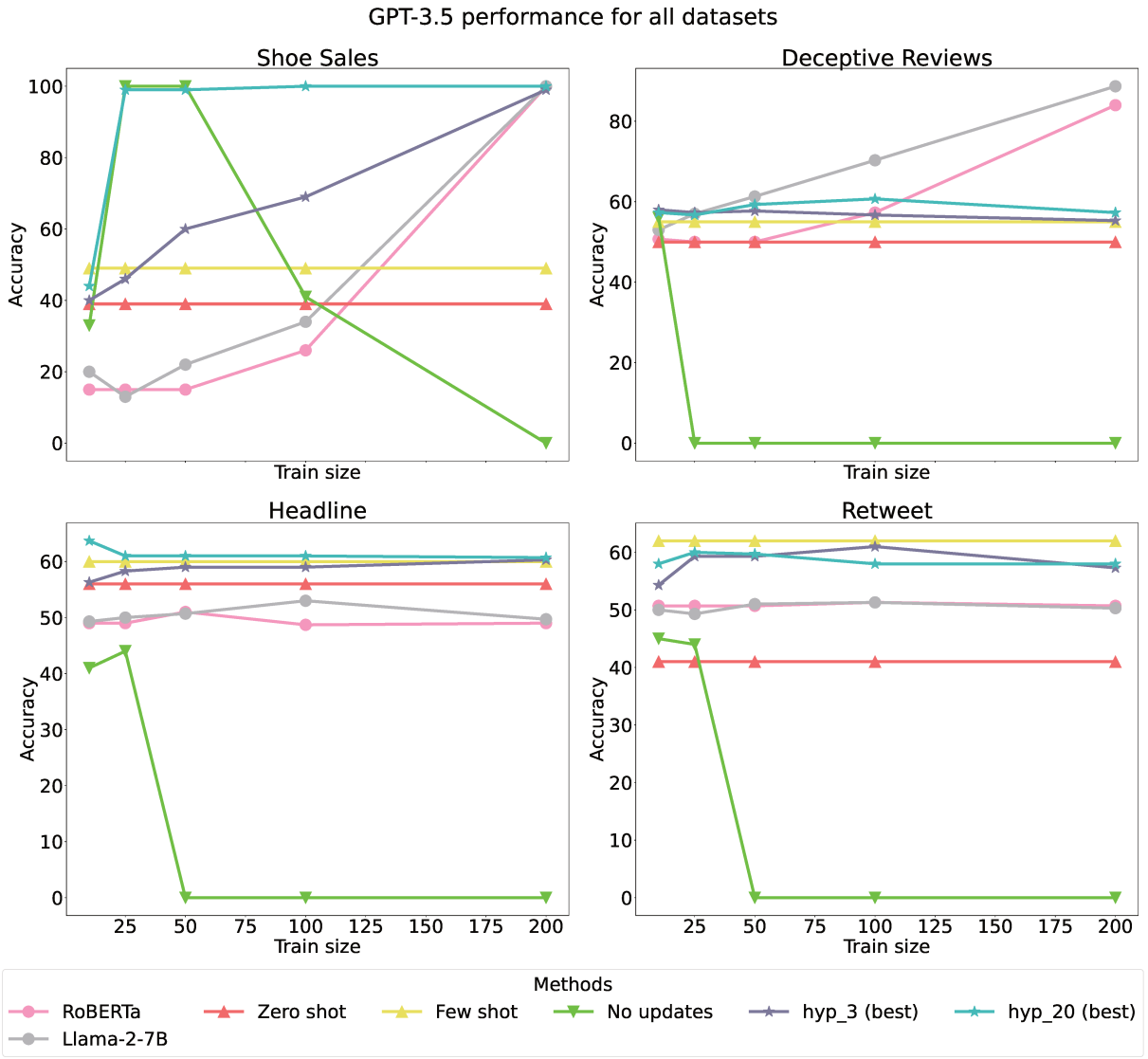


Figure 5: GPT-3.5-turbo results for baselines, **HypoGeniC** (no update), and **HypoGeniC** (best) with hypothesis bank size 3 and 20 across multiple training samples

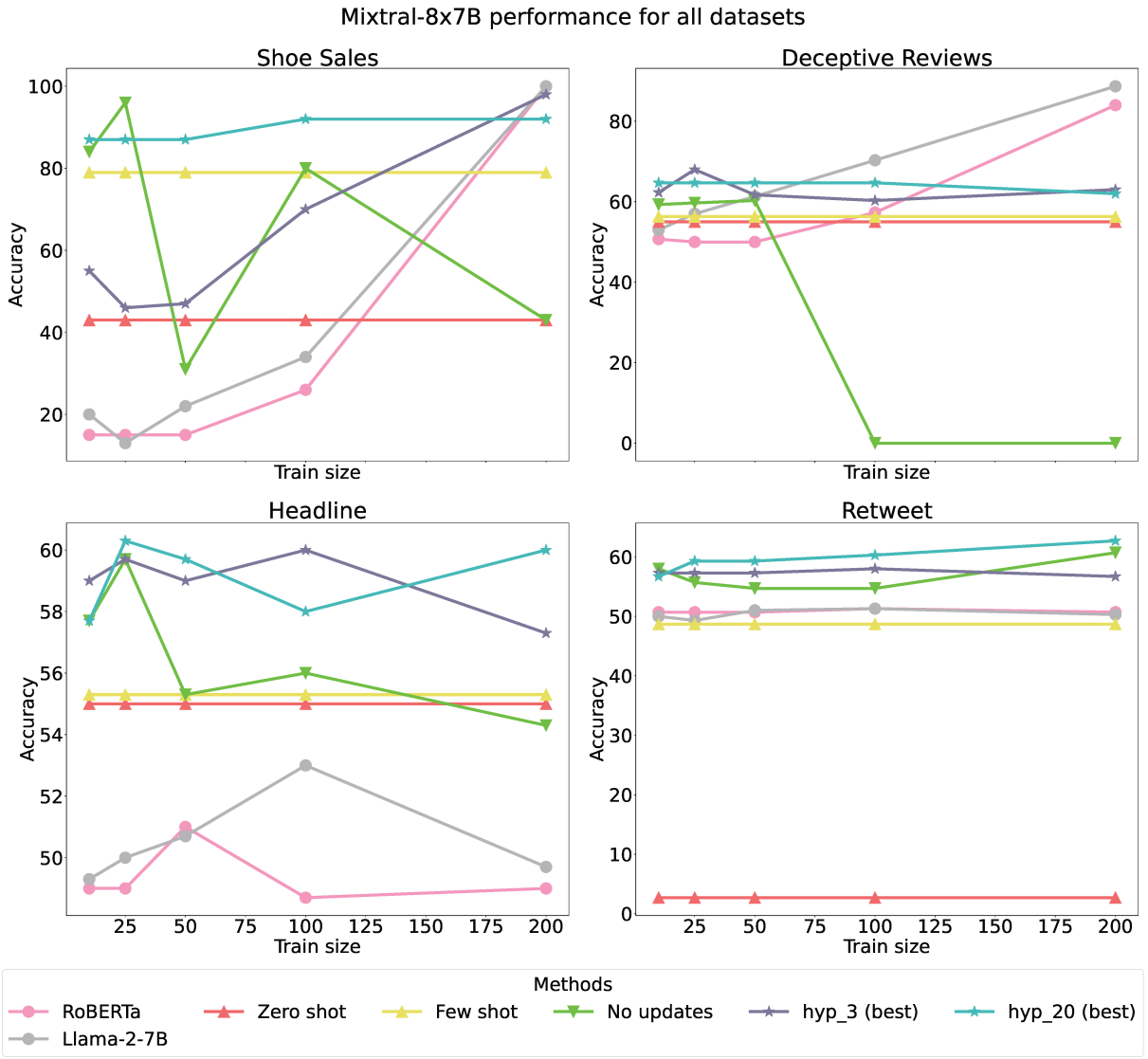


Figure 6: Mixtral results for baselines, **HypoGeniC** (no update), and **HypoGeniC** (best) with hypothesis bank size 3 and 20 across multiple training samples

Models	Methods	IND DECEPTIVE REVIEWS	OOD DECEPTIVE REVIEWS
RoBERTa (Oracle)	Train 200	84.0	73.0 (↓11.0)
	Train 1000	91.0	79.7 (↓11.3)
Llama-2-7B (Oracle)	Train 200	88.7	78.7 (↓10.0)
	Train 1000	92.3	88.7 (↓3.6)
Claude-2.1	Zero shot	31.0	27.7 (↓3.3)
	Few shot	51.0	41.7 (↓9.3)
	HypoGeniC (Best-accuracy hypothesis)	67.3	71.7 (↑4.4)
	HypoGeniC (Filter and weighted vote)	68.0	74.7 (↑6.7)
	HypoGeniC (One-step adaptive)	70.0	68.3 (↓1.7)
	HypoGeniC (Two-step adaptive)	67.7	70.7 (↑3.0)
Mixtral	Zero shot	55.0	49.7 (↓5.3)
	Few shot	56.3	49.0 (↓7.3)
	HypoGeniC (Best-accuracy hypothesis)	61.3	64.7 (↑3.4)
	HypoGeniC (Filter and weighted vote)	62.0	61.0 (↓1.0)
	HypoGeniC (One-step adaptive)	63.0	54.7 (↓8.3)
	HypoGeniC (Two-step adaptive)	61.3	64.7 (↑3.4)
GPT-3.5-turbo	Zero shot	50.0	49.0 (↓1.0)
	Few shot	55.0	52.0 (↓3.0)
	HypoGeniC (Best-accuracy hypothesis)	57.3	60.7 (↑3.4)
	HypoGeniC (Filter and weighted vote)	55.3	55.7 (↑0.4)
	HypoGeniC (One-step adaptive)	55.7	51.7 (↓4.0)
	HypoGeniC (Two-step adaptive)	54.7	59.0 (↑4.3)

Table 6: Performance of baselines and compared to our methods on the out-of-distribution deceptive reviews and DECEPTIVE REVIEWS.

Dataset	Finding	Supported/Novel
DECEPTIVE REVIEWS	Deceptive reviews contain more emotional terms.	Li et al. (2014)
	Deceptive reviews are more likely to use superlatives.	Ott et al. (2011)
	Deceptive reviews contain hearsay or information that could not have been directly experienced.	Ott et al. (2011)
	Deceptive reviews tend to be more exaggerated.	Anderson and Simester (2014)
	Truthful reviews tend to use more balanced and objective tone.	Anderson and Simester (2014)
	Truthful reviews could mention the reviewer's purpose for staying at the hotel (e.g., business trip, vacation).	Novel
	Truthful reviews would mention weddings or special occasions.	Novel
	Truthful reviews may contain information about reviewer's expectations and previous hotel experiences.	Novel
	Truthful reviews would acknowledge the reviewer's personal biases or preferences.	Novel
	Deceptive ones may present the reviewer's opinion as objective facts.	Novel
	Truthful reviews may contain reviewers' past experiences or future travel plans.	Novel
HEADLINE POPULARITY	Concreteness helps.	Sadoski et al. (2000)
	Revealing something new helps.	Banerjee and Urminsky (2021)
	Using vivid language and imagery helps.	Banerjee and Urminsky (2021)
	Headlines with high intensity of emotions would be clicked more.	Banerjee and Urminsky (2021)
	Action-oriented headlines are clicked more.	Banerjee and Urminsky (2021)
	Humorous headlines are clicked more.	Novel
	Controversial headlines are clicked more.	Novel
	Headlines that frame the content in a personal or relatable way are clicked more.	Novel
TWEET POPULARITY	Short and concise tweets are retweeted more.	Gligorić et al. (2019)
	Tweets with emotional tones are retweeted more.	Tan et al. (2014)
	Including specific details (e.g., dates, locations) are associated with more retweets.	Novel
	Including statistics and data are associated with more retweets.	Novel
	Mentioning influential individuals or organizations leads to more retweets.	Novel
	Including links to additional content (e.g., articles, videos) leads to more retweets.	Novel
	Tweets with a call to action or urgency are found to be retweeted more.	Novel

Table 7: Summary of generated hypotheses (on the real-world datasets) and whether they support existing findings or are novel.