

000
001
002
003

IN-CONTEXT LEARNING WITHOUT COPYING

004
005
006
007008 **Anonymous authors**
009
010
011
012
013
014
015
016
017
018
019
020
021
022
023
024
025
026
027
028
029
030
031
032
033
034
035
036
037
038
039
040
041
042
043
044
045
046
047
048
049
050
051
052
053040 Paper under double-blind review
041
042
043
044
045
046
047
048
049
050
051
052
053

ABSTRACT

Induction heads are attention heads that perform inductive copying by matching patterns from earlier context and copying their continuations verbatim. As models develop induction heads, they often experience a sharp drop in training loss, a phenomenon cited as evidence that induction heads may serve as a prerequisite for more complex in-context learning (ICL) capabilities. In this work, we ask whether transformers can still acquire ICL capabilities when inductive copying is suppressed. We propose HAPAX, a setting where we omit the loss contribution of any token that can be correctly predicted by induction heads. Despite a significant reduction in inductive copying, performance on abstractive ICL tasks (i.e., where the answer is not contained in the input context) remains comparable and achieves higher accuracy than the vanilla model on 12 out of 18 statistically significant tasks, even though 31.7% of tokens are omitted from the loss. Furthermore, our model achieves lower loss values on token positions that cannot be predicted correctly by induction heads. Mechanistic analysis further shows that models trained with HAPAX develop fewer and weaker induction heads but still preserve ICL capabilities. Taken together, our findings indicate that inductive copying is not essential for learning abstractive ICL mechanisms.

1 INTRODUCTION

Language modeling is fundamentally repetitive: in a coherent document, many sequences appear more than once, like “the Dursleys” in Harry Potter, or “public static void” in Java code. Words that *do* appear only once in a given text are so rare that they are given a special name by corpus linguists: *hapax legomena*, a transliteration of Ancient Greek for “said once”. But what happens if a Large Language Model (LLM) is trained such that every n -gram within its context window is a previously unseen *hapax legomena*?

In previous work, Elhage et al. (2021) showed that LLMs develop *induction heads* that perform inductive copying by matching patterns and copying them from earlier context, with Olsson et al. (2022) hypothesizing that these circuits provide a foundation for more complex in-context learning (ICL) capabilities. However, subsequent work has demonstrated that induction heads actually operate in parallel with different components (Feucht et al., 2025; Todd et al., 2024; Hendel et al., 2023) that are more causally important for performance on high-complexity ICL tasks (Yin & Steinhardt, 2025).

This motivates our central question: if we suppress a model’s ability to copy earlier sequences from context (inductive copying), can it still learn ICL capabilities? We introduce the HAPAX training regime in which repeated n -grams ($n > 1$) within a context window do not contribute to the loss. Under this constraint, the model never receives gradient signals from repeated n -grams, meaning that it can never use induction heads to predict the tokens it is being trained on. Consequently, HAPAX models’ performance on a verbatim repetition task drops by 66% relative to the vanilla model. Interestingly, we find that for the statistically significantly different tasks, HAPAX models achieve higher accuracy in 12 out of 18 and 20 out of 21 tasks across our two evaluation settings. Our findings suggest that inductive copying is causally not essential for models to develop abstractive ICL capabilities.

054

055

056

057

058

Figure 1: Demonstration of the induction circuit. Previous token heads allow each token to store which token came previously. Induction heads do a match-and-copy operation to reproduce the subsequence that appeared earlier in the context.

2 BACKGROUND

Despite being trained to do simple next-token prediction, LLMs exhibit the impressive capability to do in-context learning (ICL) (Brown et al., 2020; Chowdhery et al., 2023), where they are able to perform tasks demonstrated in-context “on-the-fly” without additional training. Olsson et al. (2022) present the first mechanistic analysis of ICL capabilities in LLMs and identify *induction circuits* as a fundamental mechanism. Induction circuits consist of three steps: (1) *previous token heads* that allow each token to store which token came before it, (2a) *induction heads* that attend to the previous token information in earlier contexts, resulting in a “prefix-matching” attention pattern, and (2b) increasing the probability of the attended token in the output. Step (2b), where the head increases the probability of the attended token, is what we refer to as inductive copying: the reproduction of token sequences that appeared earlier in the context (Figure 1). Formally, given input tokens (x_1, \dots, x_j) , induction circuits operate by searching for tokens that hold information of the current token x_j (i.e., searching x_{i+1} where $x_i = x_j, i < j$) (Elhage et al., 2021). If there is a matching x_{i+1} , the induction head increases the logit of x_{i+1} for the next prediction.

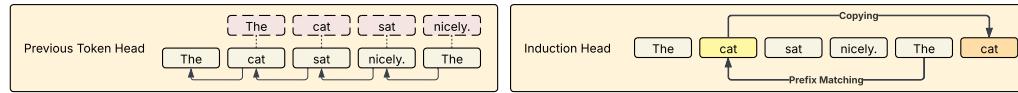
Olsson et al. (2022) observe that some of these heads are also involved in “fuzzy” copying based on semantic similarity. Feucht et al. (2025) show that models contain separate concept induction circuits that are causally more important than traditional induction circuits for “fuzzy” copying tasks (e.g., translation). In a similar vein, Yin & Steinhardt (2025) show that ablation of traditional induction heads does not damage ICL as much as ablation of *function vector* heads (Todd et al., 2024), suggesting that the latter may be more important for ICL.

Both synthetic and natural language setups have shown that the development of induction circuits is associated with rapid phase transitions as it starts to acquire prefix-matching and inductive copying capabilities (Edelman et al., 2024; Reddy, 2024; Olsson et al., 2022). After the model rapidly learns induction heads, it also experiences a general increase in ICL capabilities later in training for natural language setups. Theoretical work has identified data distributional properties that drive the emergence of in-context learning Chan et al. (2022) and shown the importance of repetition throughout data distribution for emergent behavior of large language models Zucchetti et al. (2025). Building on prior work, we examine the importance of induction heads’ ability to perform inductive copying for in context learning. Leveraging these insights, we directly manipulate the training data distribution in a natural language setting such that there is no incentive to learn inductive copying, and observe the resultant model’s ICL capabilities.

3 HAPAX

3.1 TRAINING PROTOCOL

To suppress inductive copying, we apply loss masking to create a training regime where tokens that can be correctly predicted by induction heads are excluded from the loss calculation. Previous work has used loss masking to exclude the loss contributions of certain tokens to prevent memorization of private information (Hans et al., 2024; Kosireddy & Lucas, 2025). Here, we use the same approach to remove the incentive for models to learn inductive copying. Specifically, we mask the loss contributions of token positions that contain a matching n -gram within the same context window (where $n > 1$). Single-token repetitions are not masked because they cannot be predicted by induction. Thus, the first token of any repeated n -gram is left unmasked. Importantly, masked tokens are still visible to all model components and they are only excluded from the loss computation.



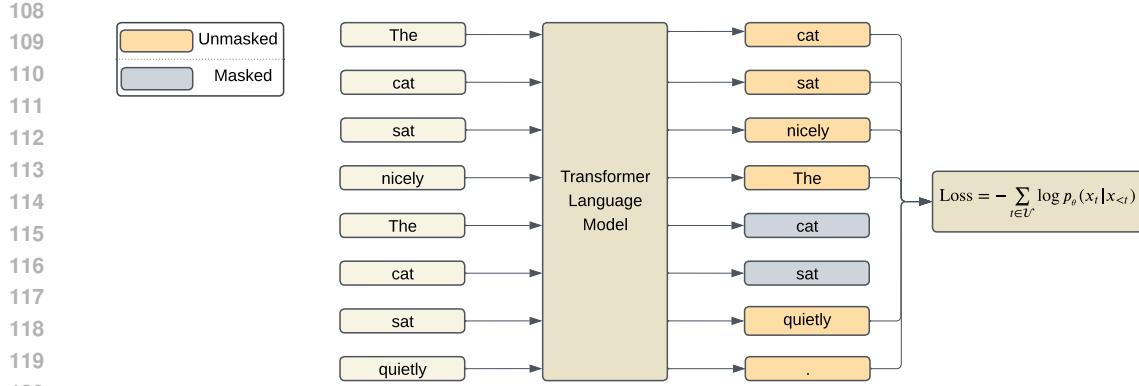


Figure 2: An overview of HAPAX training regime. To suppress inductive copying, we introduce HAPAX where positions predictable by induction heads within a context window do not contribute to the loss (gray positions). This discourages n-gram repetitions in the training distribution, allowing us to control the strength of inductive copying and observe its effect on in context learning.

Formally, let $\mathbf{x} = (x_t)_{t \in S}$ be an input sequence of tokens from vocabulary \mathcal{V} , where $S = \{1, 2, \dots, T\}$ denotes the set of all token positions and $x_t \in \mathcal{V}$ for all $t \in S$. The probability assigned by the model to the correct prediction at a position t is defined as:

$$p_{\theta}(x_t | x_{<t}) = \text{softmax}(f_{\theta}(x_{<t}))_{x_t} \quad (1)$$

where f_{θ} denotes the model with parameters θ . We define the set of masked positions as:

$$\mathcal{M} = \{j \in \mathcal{S} : \exists i < j \text{ such that } (x_{i-1}, x_i) = (x_{j-1}, x_j)\} \quad (2)$$

The set of unmasked positions is then $\mathcal{U} = \mathcal{S} \setminus \mathcal{M}$. We compute the masked loss, a negative log-likelihood, only over the unmasked positions:

$$\mathcal{L}_{\text{masked}} = -\frac{1}{|\mathcal{U}|} \sum_{t \in \mathcal{U}} \log p_{\theta}(x_t | x_{<t}) \quad (3)$$

This means that the HAPAX model is never trained on any tokens that can be correctly predicted using induction heads, removing the incentive to learn inductive copying.

We train vanilla and HAPAX 1B models from scratch using the transformers library (Wolf et al., 2020) implementation of GPT-NeoX (Andonian et al., 2023). We use the same hyperparameter and training configuration as the Pythia models (Biderman et al., 2023). We use the Pile dataset to train our models (Gao et al., 2020). We train our models for 20000 steps, which we observed empirically to be sufficient to analyze the emergence of in-context learning dynamics. We save model checkpoints every 100 training steps. The vanilla model is trained on 40B tokens whereas the HAPAX model is trained on 28B tokens due to loss masking. Further training variants that we explored but found ineffective are provided in Appendix E.

3.2 SIMILARITY-THRESHOLDED HAPAX

Certain token pairs can have a high cosine similarity in the input embedding space. As an example, the tokens “National” and “ National” (with a leading space) have a cosine similarity of 0.84 in the input embedding space of the vanilla model. Although they are not exactly the same token, the model can still acquire a high copying signal through these similar tokens. Referring back to Equation 2, we modify the exact matching with a thresholded matching logic in order to create a stricter suppression. We mask a position if the corresponding tokens have a cosine similarity higher than τ :

$$\mathcal{M} = \{j \in \mathcal{S} : \exists i < j \text{ such that } S_{\text{cos}}(e_{x_{i-1}}, e_{x_{j-1}}) > \tau \text{ and } S_{\text{cos}}(e_{x_i}, e_{x_j}) > \tau\} \quad (4)$$

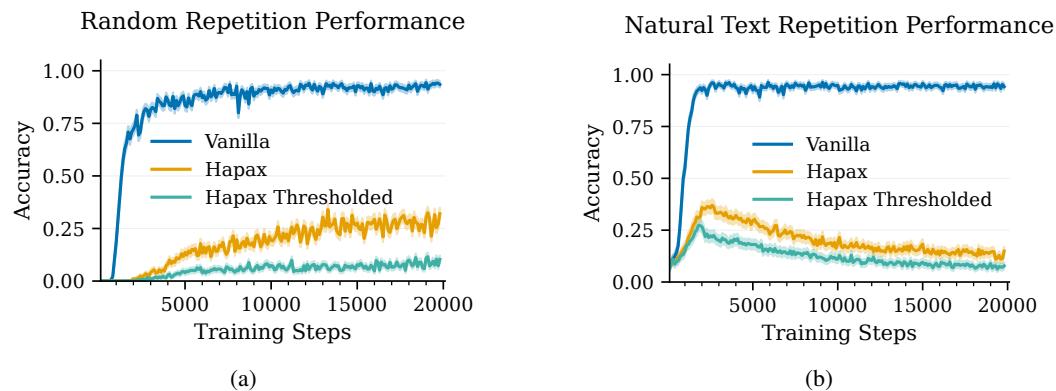
where e_{x_i} refers to the embedding for the token x_i . We choose $\tau = 0.3$. Consequently, the model is trained on 19B tokens. See Appendix B for details.

162 4 EFFECTS OF HAPAX ON ICL
163
164

165 We now test the ICL capabilities of HAPAX. In the following sections we verify that the HAPAX
166 methodology is able to suppress inductive copying capabilities. We observe a reduction for *extractive*
167 ICL task performance as described by Todd et al. (2024). An extractive task is a few-shot task
168 where the model must directly extract the answer from the input, (e.g., “foil car purple : foil, pen
169 cloud window :” → “pen”). However, when we analyze *abstractive* tasks (Todd et al., 2024), which
170 require generating new answers rather than copying (e.g., “Greece : Athens, China : Beijing, Egypt”
171 → “Cairo”), we observe that HAPAX preserves abstractive mechanisms and achieves higher accu-
172 racy on 12 out of 18 statistically significant tasks. These results provide evidence that models can
173 learn abstractive ICL without relying on inductive copying, and the incentive to not repeat previous
174 tokens might be beneficial for a subset of these tasks. In Appendix C, we complement these findings
175 with a small evaluation on the broader effect of HAPAX training on natural text generation, showing
176 an increase in natural language fluency.

177
178 4.1 SUPRESSION OF INDUCTIVE COPYING
179

180 We first measure random repetition performance: the model is given 1000 sequences of random
181 repeated tokens $r_1 r_2 \dots r_s r_1 r_2 \dots r_{s-1}$ and is expected to predict r_s . This synthetic task does not
182 occur in natural language but is solvable through induction heads and the same sequence is also
183 used to identify prefix-matching attention patterns Nanda & Bloom (2022). Figure 3a demonstrates
184 that HAPAX causes a major drop in random repetition performance. The HAPAX model experiences
185 a 66% drop and Thresholded-HAPAX experiences an 89% drop in accuracy relative to the vanilla
186 model at the end of training. We also evaluate performance on natural text repetition in Figure 3b,
187 where the models are given 1000 sequences of the form $r_1 r_2 \dots r_s r_1 r_2 \dots r_{s-1}$ with $r_1 r_2 \dots r_s$
188 taken from whole sequences in the WikiText dataset Merity et al. (2017). Natural text repetition
189 accuracy decreases over time, suggesting the model’s increased incentive to not repeat natural text.
190 The initial increase of natural text repetition is due to the model’s increased language modeling
191 capacity, as a non-repeated sequence $r_1 r_2 \dots r_{s-1}$ has the same initial accuracy for the HAPAX
192 model (Figure A9).
193



207 Figure 3: Repetition performance in both random token and natural text settings. (a) HAPAX models
208 struggle with repeating random sequences of tokens, a task that is solvable with induction circuits.
209 (b) For natural text repetition, HAPAX models actively suppress copying as training progresses.
210 Accuracy is measured over 1000 randomly generated samples with $s = 25$.

211
212
213 We evaluate HAPAX on 28 extractive tasks from Todd et al. (2024) and find statistically significant
214 changes on 19, with 17 showing reduced performance (Table A1). The results confirm that the
215 HAPAX training regime effectively reduces inductive copying and repetition. We present the results
for Thresholded-HAPAX in Appendix A.

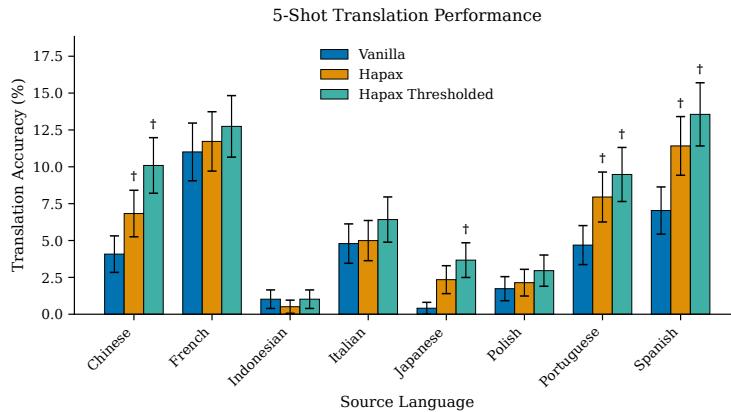


Figure 4: Word-level translation performance across all 3 models. We observe that both HAPAX and Thresholded-HAPAX increases the translation performance. This indicates that reduction in inductive copying does not hinder the model’s ability to learn translation. The translations are from a given source language into English in order to keep the number of predicted tokens same across different languages. Error bars indicate 95% binomial normal-approximation CIs. † denotes statistical significant difference ($p < 0.05$) with vanilla model accuracy according to McNemar’s test.

4.2 PERFORMANCE PRESERVATION FOR ABSTRACTIVE ICL

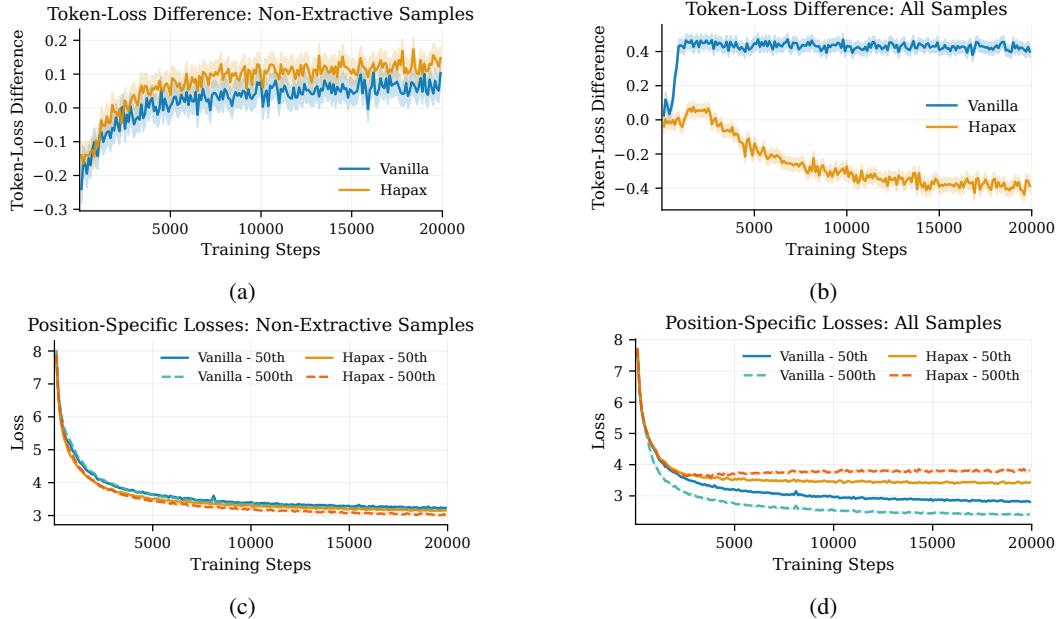
We now continue our evaluation on abstractive tasks, where the model needs to generate novel information not contained in the context. We evaluate on 23 abstractive tasks from Todd et al. (2024) (e.g., Capitalize First Letter) and 8 word-level translation tasks. All tasks are evaluated in a 5-shot setting. As shown in Figure 4 and Table 1, the model on average maintains its performance. Of the 23 abstractive tasks, HAPAX achieves higher accuracy on 9 of the 15 tasks with statistically significant differences, and is at least comparable to the vanilla model on 17 tasks overall. When we include the translation tasks, HAPAX achieves higher accuracy on 12 out of 18 significant tasks. We note that when using HAPAX, 31.7% of the tokens are masked and the preservation of performance is visible despite being trained on 31.7% fewer tokens. Thresholded-HAPAX performs worse for majority of abstractive tasks (Table A2) than vanilla model as the masking is much stricter with 52.5% of the tokens being masked. Interestingly, Thresholded-HAPAX achieves higher accuracy than HAPAX and the vanilla model on 7 of 8 tasks, with 4 statistically significant. We attribute the contrasting trends between the translation task and other abstractive tasks under Thresholded-HAPAX to our choice of threshold. A cosine similarity threshold of 0.3 masks many same-language tokens, while cross-language tokens typically fall below this threshold and therefore contribute more training signal. See Appendix B for a discussion on the choice of threshold.

Additionally, for the abstractive tasks from Todd et al. (2024), we observed that some tasks have small label spaces, causing the target token to frequently appear in the 5-shot context. To ensure the vanilla model is not matching the context’s label distribution, we reran the abstractive evaluation using only few-shot examples that exclude the target token. Under this control, the HAPAX model achieves higher accuracy than the vanilla model on 17 out of 18 significant tasks (Table A4). THRESHOLDED-HAPAX also improves, achieving higher accuracy for 10 out of 22 significant tasks, a substantial improvement over the previous results (Table A5). These suggest that the performance of the vanilla model in abstractive tasks sometimes resulted from distributional copying of labels from the context.

Overall, if abstractive ICL mechanisms were fundamentally dependent on inductive copying, we would expect performance degradation across most tasks. However, our results do not show such degradation. The disincentive of our HAPAX model to learn exact repetition has not caused significant damage, and has even increased performance for a subset of these tasks, despite being trained on much fewer tokens.

270
 271 Table 1: Performance comparison on **abstractive** in-context learning tasks (5-Shot). Values show
 272 accuracy \pm 95% CI margin. Bold indicates higher performance; \dagger denotes statistical significance
 273 ($p < 0.05$) according to McNemar’s test.

Task	Vanilla (%)	HAPAX (%)	Task	Vanilla (%)	HAPAX (%)
Capitalize First Letter	37.3 ± 3.3	$68.3 \pm 3.2^\dagger$	Singular-Plural	67.8 ± 6.4	$80.5 \pm 5.4^\dagger$
Country-Capital	31.5 ± 6.5	$43.7 \pm 6.9^\dagger$	AG News	$32.9 \pm 1.3^\dagger$	8.9 ± 0.8
Lowercase First Letter	72.2 ± 3.1	69.0 ± 3.2	Synonym	1.6 ± 0.5	$2.6 \pm 0.6^\dagger$
CommonsenseQA	$19.8 \pm 1.1^\dagger$	5.9 ± 0.7	Capitalize Second Letter	$11.7 \pm 2.2^\dagger$	2.2 ± 1.0
Landmark-Country	32.9 ± 3.2	$36.5 \pm 3.3^\dagger$	Product-Company	20.1 ± 3.4	19.2 ± 3.4
Capitalize Last Letter	$9.7 \pm 2.0^\dagger$	3.7 ± 1.3	Antonym	0.8 ± 0.3	$2.3 \pm 0.6^\dagger$
Lowercase Last Letter	6.8 ± 1.7	7.1 ± 1.8	Word Length	7.9 ± 1.8	6.9 ± 1.7
Next Item	12.0 ± 4.2	$31.1 \pm 6.0^\dagger$	National Parks	20.4 ± 3.7	23.5 ± 3.9
Country-Currency	29.9 ± 6.4	32.5 ± 6.5	Capitalize (Full Word)	$76.9 \pm 2.9^\dagger$	58.2 ± 3.4
Present-Past	54.3 ± 5.7	$68.6 \pm 5.3^\dagger$	Previous Item	5.8 ± 3.0	8.0 ± 3.5
Park-Country	12.4 ± 2.4	$17.0 \pm 2.7^\dagger$	Next Capital Letter	6.4 ± 1.7	5.9 ± 1.6
Sentiment	$65.3 \pm 2.7^\dagger$	17.4 ± 2.2			



306 Figure 5: Comparison of token-loss metrics and positional loss values across conditions. The HA-
 307 PAX model exhibits lower loss values and better TLD scores for token positions that cannot be
 308 predicted by induction. However, it receives a negative score when we consider all samples. This
 309 indicates that the loss-dependent metrics mostly capture the gains from exact copying strategy but
 310 does have implications for non-extractive samples. The metrics are calculated on randomly sampled
 311 data from the validation dataset.

4.3 IN-CONTEXT LEARNING BEYOND N-GRAM COPYING

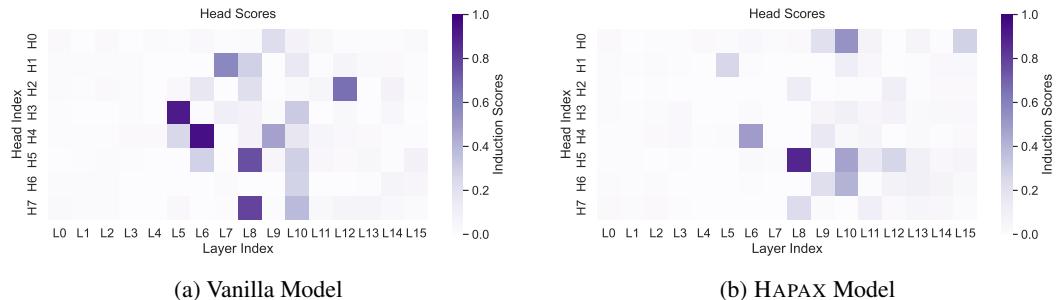
315 Building on the results from Sections 4.1 and 4.2, which suggest that models do not depend on
 316 inductive copying to acquire more abstract ICL capabilities, we now analyze loss values of HAPAX
 317 and vanilla models throughout training. We use the token-loss difference metric (termed by Yin &
 318 Steinhardt (2025) to understand general ICL capabilities. This metric is defined by the differences
 319 of the cross-entropy loss of arbitrary token positions, conventionally using the 500th and 50th token
 320 positions. Formally, let $L_t = -\log p_\theta(x_t | x_{<t})$ denote the cross-entropy loss at token position t .
 321 The token-loss difference (TLD) between the 50th and 500th tokens is then given by $\Delta TLD = L_{50} -$
 322 L_{500} . Intuitively, token-loss difference measures improvement across increasing token positions. If
 323 the loss at token 500 is lower and $TLD > 0$, it shows that the model’s predictions improved with
 324 increasing context. Yin & Steinhardt (2025) provided evidence that the metric is strongly influenced

324 by induction heads but does not correlate well with in-context learning task performance. Figure 5b
 325 shows that the HAPAX model experiences a significant drop in token-loss difference, getting worse
 326 over time. As HAPAX model performs worse at exact copying tasks but maintains its performance
 327 on abstract copying tasks, our results demonstrate that the sudden increase in token-loss difference
 328 metric for vanilla model is indicative of how the model learns inductive copying, but should not
 329 be taken as a measure of abstractive learning capability, which is consistent with observations from
 330 Lv et al. (2025). In natural language, n-grams are often repeated, and the token-loss difference
 331 metric is primarily affected by these exact copying instances. Because our loss masking removes
 332 31.7% of tokens, it implies that exactly 31.7% of positions in the data are those that induction could
 333 have correctly predicted through copying. This statistic is implied by the extension of Zipf’s Law
 334 to n -grams Ha et al. (2009). All these results provide evidence that the exact copying capability
 335 and the phase shifts in loss-dependent metrics gained by induction heads can be explained primarily
 336 by the distributional attributes of natural language, but it does not have implications for abstractive
 337 capabilities.

338 To investigate this hypothesis further, we propose using samples where neither the 500th nor the 50th
 339 token can be predicted correctly with inductive copying. With this modification, we will be able
 340 to understand the improvement of ICL for tokens that cannot be predicted with inductive copying.
 341 Figure 5a shows that, contrary to the regular token-loss difference metric, the HAPAX model has a
 342 slightly higher token-loss difference, which suggests that it can leverage context better for non-exact
 343 copying instances and aligns with results from Section 4.2. We also observe that the model’s ability
 344 to leverage context for non-exact matching tokens does not exhibit a phase shift but rather improves
 345 gradually across training steps. Figure 5c shows that not only the difference but also the individual
 346 losses at these positions are lower in the loss masked model. This demonstrates that learning to
 347 leverage context for non-exact copying instances (as calculated by the TLD score on non-extractive
 348 samples) has improved, despite the reduction in inductive copying.

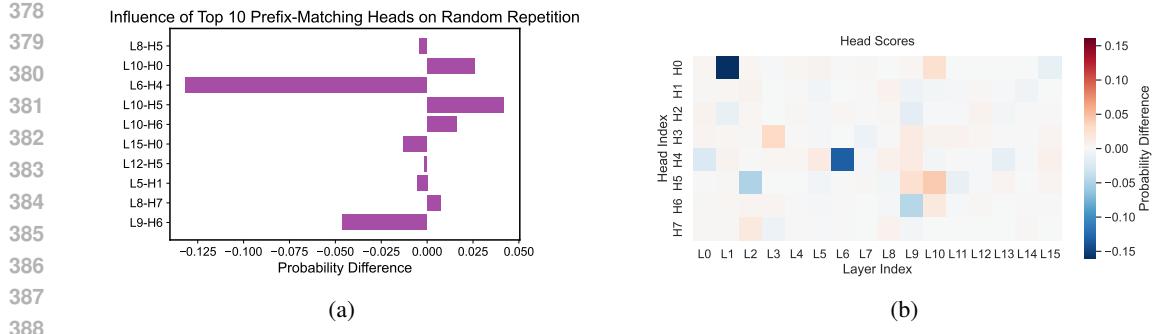
349 5 MECHANISTIC INVESTIGATION OF INDUCTION HEADS

350 5.1 INFLUENCE OF PREFIX MATCHING HEADS ON COPYING



351
 352 Figure 6: Prefix Matching Scores across Vanilla and HAPAX models. HAPAX model has fewer heads
 353 that have a strong prefix-matching score.

354
 355 With inductive copying suppressed, we next investigate mechanistically how induction heads are af-
 356 fected. We analyze the attention patterns of attention heads for the vanilla and HAPAX models using
 357 the random repetition sequence discussed in Section 4.1. To obtain prefix matching score over a
 358 sequence $x = (r_1 r_2 \dots r_s r_1 r_2 \dots r_{s-1})$, we calculate $\text{PrefixMatching}(l, h) = \frac{1}{s-1} \sum_{i=1}^{s-1} A_{s+i, i+1}^{(l, h)}$
 359 over 1000 samples, where $s = 25$ and $A^{l, h}$ is the attention map of attention head at layer l and
 360 index h Nanda & Bloom (2022). HAPAX has fewer attention heads that strongly display the prefix-
 361 matching pattern commonly associated with induction heads (Figures 6a and 6b). In the vanilla
 362 model, the top 10 prefix-matching heads achieve an average score of 61%, whereas in HAPAX this
 363 average drops to 40%. Also, there is only a single attention head (L8H5) that attends with a near
 364 100% score. As noted in Section 4.1, other head types can also exhibit a similar pattern. We there-
 365 fore analyze the prefix-matching heads to quantify their contribution on inductive copying.



392

393

394 Figure 7: Influence of individual attention heads for inductive copying. Different from the vanilla

395 model, many of the top 10 prefix-matching heads are negatively influencing copying, as shown in

396 (a).

397

398 Using the same setup, we conduct ablation studies to understand the causal impact of each individual

399 attention head for the expected prediction r_s . Mean ablation is a method that replaces the activation

400 of a model component with its average activation from a reference distribution (Wang et al., 2023).

401 For each head, we compute the mean activation over 10000 samples from the Pile dataset that the

402 model was trained on (Gao et al., 2020). We only mean ablate the activations of the last token

403 position in order to isolate the behavior of induction heads (see Appendix A.2). We look at the

404 probability difference of the target token r_s before and after ablation, computing $p_{clean}(r_s|x) -$

405 $p_{ablated}(r_s|x)$

406 Figure 7b shows the probability differences for each head after mean ablating. To get a clearer

407 view of how prefix matching determines copying behavior, we display in Figure 7a the probability

408 difference scores of the top 10 heads that have the highest prefix matching scores with their rankings

409 preserved. We observe from Figure 7a that out of the top 10 prefix matching heads, 6 of the heads

410 negatively influence the probability assigned to correct token, meaning that they functionally behave

411 closer to an anti-induction head (McDougall et al., 2024; Olsson et al., 2022). Despite many of the

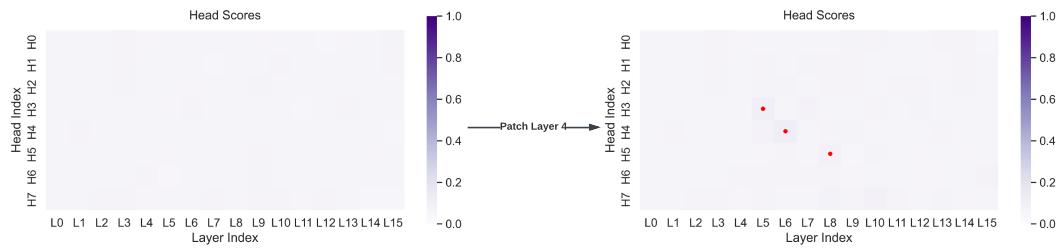
412 top 10 prefix-matching heads negatively influencing prediction, abstract ICL capabilities remain

413 intact. This suggests that learning abstractive ICL relies less on inductive copying. A reduction in

414 inductive copying and induction heads did not compromise the learning of abstractive capabilities

415 for the HAPAX model.

416 5.2 CROSS-CHECKPOINT PATCHING EXPERIMENTS



422 Figure 8: Induction head scores of the randomly initialized vanilla model before and after patching

423 layer L4 of the vanilla model at step 5000, which contains the first previous-token heads (Figure

424 A11). The maximum induction score on a randomly repeated sequence is 0.076, while patching

425 layer L4 raises the maximum to 0.122. Heads marked with red dots are the top three prefix-matching

426 heads which also rank among the top prefix-matching heads in the vanilla model’s final checkpoint,

427 showing that some heads are biased toward prefix matching from initialization and once previous

428 token heads are formed, certain heads are naturally going to attend to the flowing information.

429

430 With HAPAX training, we obtained a model that does not benefit from repetition. However, our

431 data distribution plausibly does not imply anything about the existence of previous token heads, and

432 they might still be helpful for tasks such as detokenization (Lad et al., 2025; Gurnee et al., 2023;

432 Feucht et al., 2024; Kaplan et al., 2025). In this section, we conduct another set of experiments to
 433 ascertain the influence of previous token heads on the formation of induction heads. If models must
 434 develop previous token heads for reasons other than learning induction circuits, heads in later layers
 435 may naturally develop prefix-matching attention patterns as they attend to this information, which
 436 is why we might still see reduced but non-zero inductive copying behavior. Here, we find that even
 437 *randomly initialized* heads at later layers will attend to previous token information, suggesting that
 438 prefix-matching patterns can form as a direct result of the presence of previous token information.

439 We analyze two models: our vanilla model at step 5000, and the same model at step 0 (random
 440 initialization). At step 5000, our vanilla model has previous token heads at layer L4, but no induction
 441 heads appear in layer L4 or earlier (Figure A11). We hypothesize that when this previous token
 442 information is present, randomly initialized heads in later layers will begin to exhibit prefix-matching
 443 attention patterns. Therefore, we apply activation patching by patching the outputs of layer L4 from
 444 step 5000 into layer L4 at step 0, and observe whether heads in later layers attend to this previous
 445 token information despite being randomly initialized.

446 Figure 8 shows that once we patch the outputs of layer L4 at step 5000 into the randomly-initialized
 447 model, the model’s highest induction score increases to 0.122. (Before patching, the randomly-
 448 initialized model had a maximum prefix matching score of 0.076.) All heads highlighted with red
 449 dot for this experiment also rank among the top 5 prefix matching heads at the final checkpoint of
 450 the vanilla model (Figure 6a), indicating that these heads were biased towards prefix matching from
 451 initialization. A similar pattern holds for the HAPAX model, where 2 out of 3 top prefix-matching
 452 heads from its final checkpoint also have increased induction scores in our cross-checkpoint patch-
 453 ing experiment (L8H5, L6H4). Both the HAPAX and vanilla model start training from the same
 454 random initialization seed, which can explain why we are seeing this overlap. These results suggest
 455 that induction head-like attention patterns can form quite easily once previous token information
 456 is present, possibly explaining how the HAPAX model is still able to learn such attention patterns
 457 despite never being trained on token positions that can be predicted by induction heads.

460 6 RELATED WORK

461
 462 **Attention Heads and ICL** Early work identified induction circuits responsible for inductive copying
 463 (Elhage et al., 2021) and analyzed their influence on ICL (Olsson et al., 2022; Crosbie & Shutova,
 464 2025), hypothesizing they are the basis of in-context learning. Subsequent studies found various
 465 attention heads important for ICL, including function vector heads that trigger tasks (Todd et al.,
 466 2024), concept induction heads that copy lexical units (Feucht et al., 2025), semantic induction
 467 heads extracting relations (Ren et al., 2024), symbolic induction heads inducing over abstract vari-
 468 ables (Yang et al., 2025), and n-gram generalizations of induction heads (Akyürek et al., 2024).
 469 Prior work also noted correlations between task-specific heads and canonical induction heads, ques-
 470 tioning whether the latter are necessary for learning ICL-related attention heads. We show that
 471 despite weaker induction heads and reduced copying capacity, models maintain robust abstract ICL
 472 capabilities that do not require exact copying.

473 **Training Dynamics and Induction Heads.** Theoretic work on synthetic setups has provided in-
 474 sights into induction head development and training dynamics. Several studies analyze phase transi-
 475 tions where Singh et al. (2024) use clamping to study subcomponents of induction heads and
 476 their effect on phase changes. Chan et al. (2022) shows how skewed Zipfian distributions lead to
 477 the emergence of ICL. Bietti et al. (2023) demonstrates the evolution from global bigram statis-
 478 tics to induction head solutions over training, and Chen et al. (2024); Edelman et al. (2024) further
 479 analyze convergence to induction-like solutions in Markov chain data with Edelman et al. (2024)
 480 demonstrating the different phases where the model learns more complex strategies with more train-
 481 ing. Minegishi et al. (2025) studies a meta-learning setting to investigate mechanisms in non-exact
 482 copying scenarios. In our work, we instead focus on a natural language setting and create a data
 483 distribution where induction heads can never predict the correct next token.

484 **Loss Masking.** Prior work incorporated loss masking strategies to prevent memorization for infor-
 485 mation of interest (Hans et al., 2024; Kosireddy & Lucas, 2025). We incorporate a loss masking
 486 strategy to create a distribution which allows us to suppress induction behavior throughout training.

486 **Repetition.** Prior work has studied repetition from the perspective of data distribution and model
 487 outputs. Zucchet et al. (2025) analyzes how training data repetition speeds up emergent behavior in
 488 language models. Several works analyze the “repetition curse”, with some finding induction heads
 489 to be causing such behavior (Hiraoka & Inui, 2025; Wang et al., 2025; Yao et al., 2025). Welleck
 490 et al. (2020) presents unlikelihood loss training to mitigate repetitions in the model. As repetition is
 491 closely tied to induction heads, our HAPAX training regime discourages repeated sequences, aiming
 492 to suppress inductive copying and, as a result, repetition.

494 7 CONCLUSION

496 In this work, we introduce the HAPAX training regime, which removes the incentive for the in-
 497 ductive copying strategy typically learned by induction heads early in training. We find that while
 498 HAPAX models struggle more with inductive copying and develop fewer induction heads, their ab-
 499 stractive ICL capabilities are preserved, and in some cases even improved. Our analysis shows that
 500 inductive copying is not essential for learning abstractive ICL mechanisms. We also observe weaker
 501 induction circuits, which suggests their role in learning abstractive ICL capabilities could be more
 502 limited than previously assumed. We observe that reducing repetition incentives does not adversely
 503 affect the abstractive behavior and has benefits for quality of the generation. More broadly, our
 504 methodology offers a systematic way to analyze model learning dynamics and can be adapted to
 505 explore how different attributes of the data distribution may shape learning dynamics.

507 ETHICS STATEMENT

509 This paper aims to advance the foundational understanding of in-context learning mechanisms.
 510 While such research may influence future model development and deployment, we cannot mean-
 511 ingfully anticipate these downstream impacts within the scope of this work.

513 REPRODUCIBILITY STATEMENT

515 We release training and evaluation code and data to enable replication of our findings. This includes
 516 environments, configurations (covering tokenizer, architecture, and optimizer settings), and specific
 517 seeds used for data shuffling and evaluation. The code also implements our HAPAX objective, in-
 518 cluding similarity-thresholded masking, and provides evaluation prompts and mechanistic analysis.
 519 For patching experiments, we used NNSIGHT (Fiotto-Kaufman et al., 2025).

522 REFERENCES

523 Ekin Akyürek, Bailin Wang, Yoon Kim, and Jacob Andreas. In-context language learning: architec-
 524 tures and algorithms. In *Proceedings of the 41st International Conference on Machine Learning*,
 525 ICML’24. JMLR.org, 2024.

527 Alex Andonian, Quentin Anthony, Stella Biderman, Sid Black, Preetham Gali, Leo Gao, Eric
 528 Hallahan, Josh Levy-Kramer, Connor Leahy, Lucas Nestler, Kip Parker, Michael Pieler, Jason
 529 Phang, Shivanshu Purohit, Hailey Schoelkopf, Dashiell Stander, Tri Songz, Curt Tigges, Ben-
 530 jamin Thérien, Phil Wang, and Samuel Weinbach. GPT-NeoX: Large Scale Autoregressive Lan-
 531 guage Modeling in PyTorch, 9 2023. URL <https://www.github.com/eleutherai/gpt-neox>.

533 Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O’Brien, Eric
 534 Hallahan, Mohammad Aftab Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al.
 535 Pythia: A suite for analyzing large language models across training and scaling. In *International
 536 Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

538 Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. Birth of a
 539 transformer: A memory viewpoint. In *Thirty-seventh Conference on Neural Information Process-
 ing Systems*, 2023. URL <https://openreview.net/forum?id=3X2EbBLNsK>.

540 Steven Bird and Edward Loper. NLTK: The natural language toolkit. In *Proceedings of the ACL*
 541 *Interactive Poster and Demonstration Sessions*, pp. 214–217, Barcelona, Spain, July 2004. Association
 542 for Computational Linguistics. URL <https://aclanthology.org/P04-3031/>.

543 Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhari-
 544 wal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agar-
 545 wal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh,
 546 Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz
 547 Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec
 548 Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In
 549 H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neu-
 550 ral Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc.,
 551 2020. URL [https://proceedings.neurips.cc/paper_files/paper/2020/
 552 file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf).

553 Stephanie C.Y. Chan, Adam Santoro, Andrew K. Lampinen, Jane X. Wang, Aaditya K. Singh,
 554 Pierre H. Richemond, James L. McClelland, and Felix Hill. Data distributional properties drive
 555 emergent in-context learning in transformers. In *Proceedings of the 36th International Confer-
 556 ence on Neural Information Processing Systems*, NIPS ’22, Red Hook, NY, USA, 2022. Curran
 557 Associates Inc. ISBN 9781713871088.

558 Siyu Chen, Heejune Sheen, Tianhao Wang, and Zhuoran Yang. Unveiling induction heads: Prov-
 559 able training dynamics and feature learning in transformers. In *The Thirty-eighth Annual Confer-
 560 ence on Neural Information Processing Systems*, 2024. URL [https://openreview.net/
 561 forum?id=4fN2REs0Ma](https://openreview.net/forum?id=4fN2REs0Ma).

562 Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam
 563 Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh,
 564 Kensen Shi, Sashank Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay,
 565 Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope,
 566 James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm
 567 Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra,
 568 Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Bar-
 569 ret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick,
 570 Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica
 571 Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Bren-
 572 nan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas
 573 Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: scaling language modeling with pathways.
J. Mach. Learn. Res., 24(1), January 2023. ISSN 1532-4435.

574 Joy Crosbie and Ekaterina Shutova. Induction heads as an essential mechanism for pattern match-
 575 ing in in-context learning. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of*
 576 *the Association for Computational Linguistics: NAACL 2025*, pp. 5034–5096, Albuquerque,
 577 New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-
 578 7. doi: 10.18653/v1/2025.findings-naacl.283. URL [https://aclanthology.org/2025.
 579 findings-naacl.283/](https://aclanthology.org/2025.findings-naacl.283/).

580 Thomas G. Dietterich. Approximate statistical tests for comparing supervised classification learning
 581 algorithms. *Neural Comput.*, 10(7):1895–1923, October 1998. ISSN 0899-7667. doi: 10.1162/
 582 089976698300017197. URL <https://doi.org/10.1162/089976698300017197>.

583 Ezra Edelman, Nikolaos Tsilivis, Benjamin L. Edelman, eran malach, and Surbhi Goel. The
 584 evolution of statistical induction heads: In-context learning markov chains. In *The Thirty-
 585 eighth Annual Conference on Neural Information Processing Systems*, 2024. URL <https://openreview.net/forum?id=qaRT6QTIqJ>.

586 Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann,
 587 Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep
 588 Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt,
 589 Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and
 590 Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*,
 591 2021. <https://transformer-circuits.pub/2021/framework/index.html>.

594 Sheridan Feucht, David Atkinson, Byron Wallace, and David Bau. Token erasure as a footprint
 595 of implicit vocabulary items in llms. In *The 2024 Conference on Empirical Methods in Natural
 596 Language Processing*, 2024. URL <https://arxiv.org/abs/2406.20086>.

597 Sheridan Feucht, Eric Todd, Byron C Wallace, and David Bau. The dual-route model of induction.
 598 In *Second Conference on Language Modeling*, 2025. URL <https://openreview.net/forum?id=bNTrKqqnG9>.

600 Jaden Fried Fiotto-Kaufman, Alexander Russell Loftus, Eric Todd, Jannik Brinkmann, Koyena Pal,
 601 Dmitrii Troitskii, Michael Ripa, Adam Belfki, Can Rager, Caden Juang, Aaron Mueller, Samuel
 602 Marks, Arnab Sen Sharma, Francesca Lucchetti, Nikhil Prakash, Carla E. Brodley, Arjun Guha,
 603 Jonathan Bell, Byron C Wallace, and David Bau. NNsight and NDIF: Democratizing access to
 604 open-weight foundation model internals. In *The Thirteenth International Conference on Learning
 605 Representations*, 2025. URL <https://openreview.net/forum?id=MxbEiFRf39>.

606 Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason
 607 Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text
 608 for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

609 Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris
 610 Bertsimas. Finding neurons in a haystack: Case studies with sparse probing, 2023. URL
 611 <https://arxiv.org/abs/2305.01610>.

612 Le Quan Ha, Philip Hanna, Ji Ming, and F J Smith. Extending zipf’s law to n-grams for large
 613 corpora. *Artif. Intell. Rev.*, 32(1-4):101–113, December 2009.

614 Abhimanyu Hans, John Kirchenbauer, Yuxin Wen, Neel Jain, Hamid Kazemi, Prajwal Singhania,
 615 Siddharth Singh, Gowthami Somepalli, Jonas Geiping, Abhinav Bhatele, and Tom Goldstein. Be
 616 like a goldfish, don’t memorize! mitigating memorization in generative LLMs. In *The Thirty-
 617 eighth Annual Conference on Neural Information Processing Systems*, 2024. URL <https://openreview.net/forum?id=DylSyAfmWs>.

618 Roei Hendel, Mor Geva, and Amir Globerson. In-context learning creates task vectors. In
 619 Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Compu-
 620 tational Linguistics: EMNLP 2023*, pp. 9318–9333, Singapore, December 2023. Association
 621 for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.624. URL <https://aclanthology.org/2023.findings-emnlp.624/>.

622 Tatsuya Hiraoka and Kentaro Inui. Repetition neurons: How do language models produce repeti-
 623 tions? In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference
 624 of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human
 625 Language Technologies (Volume 2: Short Papers)*, pp. 483–495, Albuquerque, New Mexico, April
 626 2025. Association for Computational Linguistics. ISBN 979-8-89176-190-2. doi: 10.18653/v1/
 627 2025.nacl-short.41. URL <https://aclanthology.org/2025.nacl-short.41/>.

628 Guy Kaplan, Matanel Oren, Yuval Reif, and Roy Schwartz. From tokens to words: On the inner
 629 lexicon of llms, 2025. URL <https://arxiv.org/abs/2410.05864>.

630 Tagore Rao Kosireddy and Evan Lucas. Empirical evaluation of loss masking to selectively prevent
 631 memorization. In Robin Jia, Eric Wallace, Yangsibo Huang, Tiago Pimentel, Pratyush Maini,
 632 Verna Dankers, Johnny Wei, and Pietro Lesni (eds.), *Proceedings of the First Workshop on Large
 633 Language Model Memorization (L2M2)*, pp. 142–149, Vienna, Austria, August 2025. Association
 634 for Computational Linguistics. ISBN 979-8-89176-278-7. doi: 10.18653/v1/2025.l2m2-1.11.
 635 URL <https://aclanthology.org/2025.l2m2-1.11/>.

636 Vedang Lad, Jin Hwa Lee, Wes Gurnee, and Max Tegmark. The remarkable robustness of llms:
 637 Stages of inference?, 2025. URL <https://arxiv.org/abs/2406.19384>.

638 Ang Lv, Ruobing Xie, Xingwu Sun, Zhanhui Kang, and Rui Yan. Language models “grok” to
 639 copy. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of
 640 the Nations of the Americas Chapter of the Association for Computational Linguistics: Human
 641 Language Technologies (Volume 2: Short Papers)*, pp. 735–741, Albuquerque, New Mexico, April
 642 2025. Association for Computational Linguistics. ISBN 979-8-89176-190-2. doi: 10.18653/v1/
 643 2025.nacl-short.61. URL <https://aclanthology.org/2025.nacl-short.61/>.

648 Callum Stuart McDougall, Arthur Conmy, Cody Rushing, Thomas McGrath, and Neel Nanda.
 649 Copy suppression: Comprehensively understanding a motif in language model attention heads.
 650 In Yonatan Belinkov, Najoung Kim, Jaap Jumelet, Hosein Mohebbi, Aaron Mueller, and Han-
 651 jie Chen (eds.), *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting*
 652 *Neural Networks for NLP*, pp. 337–363, Miami, Florida, US, November 2024. Association
 653 for Computational Linguistics. doi: 10.18653/v1/2024.blackboxnlp-1.22. URL <https://aclanthology.org/2024.blackboxnlp-1.22/>.

655 Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mix-
 656 ture models. In *International Conference on Learning Representations*, 2017. URL <https://openreview.net/forum?id=Byj72udxe>.

657 Gouki Minegishi, Hiroki Furuta, Shohei Taniguchi, Yusuke Iwasawa, and Yutaka Matsuo. Beyond
 658 induction heads: In-context meta learning induces multi-phase circuit emergence. In *Forty-second*
 659 *International Conference on Machine Learning*, 2025. URL <https://openreview.net/forum?id=Xw01vF13aV>.

660 Neel Nanda and Joseph Bloom. Transformerlens. <https://github.com/TransformerLensOrg/TransformerLens>, 2022.

661 Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan,
 662 Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli,
 663 Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane
 664 Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish,
 665 and Chris Olah. In-context learning and induction heads. *Transformer Circuits Thread*, 2022.
 666 <https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html>.

667 Gautam Reddy. The mechanistic basis of data dependence and abrupt learning in an in-context
 668 classification task. In *The Twelfth International Conference on Learning Representations*, 2024.
 669 URL <https://openreview.net/forum?id=aN4Jf6Cx69>.

670 Jie Ren, Qipeng Guo, Hang Yan, Dongrui Liu, Quanshi Zhang, Xipeng Qiu, and Dahu Lin. Iden-
 671 tifying semantic induction heads to understand in-context learning. In Lun-Wei Ku, Andre Martins,
 672 and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL*
 673 2024, pp. 6916–6932, Bangkok, Thailand, August 2024. Association for Computational Linguis-
 674 tics. doi: 10.18653/v1/2024.findings-acl.412. URL [https://aclanthology.org/2024.findings-acl.412/](https://aclanthology.org/2024.findings-acl.412).

675 Aaditya K. Singh, Ted Moskovitz, Felix Hill, Stephanie C. Y. Chan, and Andrew M. Saxe. What
 676 needs to go right for an induction head? a mechanistic study of in-context learning circuits
 677 and their formation. In *Proceedings of the 41st International Conference on Machine Learning*,
 678 ICML’24. JMLR.org, 2024.

679 Eric Todd, Millicent Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau.
 680 Function vectors in large language models. In *The Twelfth International Conference on Learning*
 681 *Representations*, 2024. URL <https://openreview.net/forum?id=AwyxtyMwaG>.

682 Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt.
 683 Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In
 684 *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=NpsVSN6o4ul>.

685 Shuxun Wang, Qingyu Yin, Chak Tou Leong, Qiang Zhang, and Linyi Yang. Induction head tox-
 686 icticity mechanistically explains repetition curse in large language models, 2025. URL <https://arxiv.org/abs/2505.13514>.

687 Maurice Weber, Daniel Y Fu, Quentin Gregory Anthony, Yonatan Oren, Shane Adams, Anton
 688 Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun,
 689 Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Re, Irina
 690 Rish, and Ce Zhang. Redpajama: an open dataset for training large language models. In *The*
 691 *Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks*
 692 *Track*, 2024. URL <https://openreview.net/forum?id=lnuXaRpww>.

702 Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston.
 703 Neural text generation with unlikelihood training. In *International Conference on Learning Rep-*
 704 *resentations*, 2020. URL <https://openreview.net/forum?id=SJeYe0NtvH>.

705
 706 Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi,
 707 Pierrick Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick
 708 von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger,
 709 Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural
 710 language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural*
 711 *Language Processing: System Demonstrations*, pp. 38–45. Association for Computational Lin-
 712 *guistics*, 2020. URL <https://www.aclweb.org/anthology/2020.emnlp-demos.6>.

713
 714
 715 Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christo-
 716 pher D Manning, and Christopher Potts. Axbench: Steering LLMs? even simple baselines out-
 717 perform sparse autoencoders. In *Forty-second International Conference on Machine Learning*,
 718 2025. URL <https://openreview.net/forum?id=K2CckZjNy0>.

719
 720 Yukang Yang, Declan Iain Campbell, Kaixuan Huang, Mengdi Wang, Jonathan D. Cohen, and Tay-
 721 lor Whittington Webb. Emergent symbolic mechanisms support abstract reasoning in large lan-
 722 guage models. In *Forty-second International Conference on Machine Learning*, 2025. URL
 723 <https://openreview.net/forum?id=y1SnRPDWx4>.

724
 725 Junchi Yao, Shu Yang, Jianhua Xu, Lijie Hu, Mengdi Li, and Di Wang. Understanding the repeat
 726 curse in large language models from a feature perspective. In Wanxiang Che, Joyce Nabende,
 727 Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Com-*
 728 *putational Linguistics: ACL 2025*, pp. 7787–7815, Vienna, Austria, July 2025. Association for
 729 Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.406.
 730 URL <https://aclanthology.org/2025.findings-acl.406/>.

731
 732 Kayo Yin and Jacob Steinhardt. Which attention heads matter for in-context learning? In *Forty-*
 733 *second International Conference on Machine Learning*, 2025. URL <https://openreview.net/forum?id=C7XmEByCFv>.

734
 735
 736 Nicolas Zucchet, Francesco D’Angelo, Andrew Kyle Lampinen, and Stephanie C.Y. Chan. The
 737 emergence of sparse attention: impact of data distribution and benefits of repetition. In *The*
 738 *Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL <https://openreview.net/forum?id=jMhRbV47pS>.

742 A ADDITIONAL MODEL RESULTS

744 A.1 ICL TASK RESULTS

746 A.2 ABLATION STUDIES

748 While conducting mean ablations, we only ablate the last token position of the attention heads. We
 749 choose the last token to avoid disrupting the influence of the rest of the circuit, mainly the previous
 750 token heads. Given a sequence $r_1 r_2 \dots r_s r_1 r_2 \dots r_{s-1}$ the previous token head that is going to hold
 751 information for prefix matching is going to be stored at token position r_s , which is the end of the first
 752 half of the repetition. If we were to ablate all token positions, we would also disrupt the information
 753 flow from previous token heads, which would mistakenly lead us to identify such heads as induction
 754 heads. We also include the figures of vanilla and Thresholded-HAPAX for comparison with the
 755 HAPAX model. We chose to use probability difference metric as it correlated most accurately with
 task accuracy.

756
757
758
759
760Table A1: Performance comparison on **extractive** in-context learning tasks (5-Shot) for Vanilla and HAPAX models. Values show accuracy \pm 95% CI margin. Bold indicates higher performance; \dagger denotes statistically significance ($p < 0.05$) according to McNemar’s test.

Task	Vanilla (%)	HAPAX (%)	Task	Vanilla (%)	HAPAX (%)
Choose First of 3	95.0 \pm 1.4\dagger	30.7 \pm 2.9	Choose First of 5	95.9 \pm 1.2\dagger	27.0 \pm 2.8
Choose Middle of 3	40.2 \pm 3.0\dagger	21.5 \pm 2.5	Choose Middle of 5	26.7 \pm 2.7\dagger	7.4 \pm 1.6
Choose Last of 3	66.1 \pm 2.9\dagger	53.6 \pm 3.1	Choose Last of 5	52.2 \pm 3.1	58.7 \pm 3.1\dagger
Alphabetically First (3)	34.4 \pm 2.9\dagger	28.1 \pm 2.8	Alphabetically First (5)	19.9 \pm 2.5\dagger	14.5 \pm 2.2
Alphabetically Last (3)	32.3 \pm 2.9\dagger	24.5 \pm 2.7	Alphabetically Last (5)	18.5 \pm 2.4\dagger	13.0 \pm 2.1
CoNLL: Person	73.2 \pm 1.5\dagger	25.8 \pm 1.4	CoNLL: Location	59.4 \pm 1.4\dagger	17.5 \pm 1.1
CoNLL: Organization	57.8 \pm 1.6\dagger	21.2 \pm 1.3	Animal vs Object (3)	48.4 \pm 3.1\dagger	27.3 \pm 2.8
Animal vs Object (5)	36.3 \pm 3.0\dagger	16.3 \pm 2.3	Fruit vs Animal (3)	45.1 \pm 3.1\dagger	21.5 \pm 2.5
Fruit vs Animal (5)	28.1 \pm 2.8\dagger	9.9 \pm 1.9	Color vs Animal (3)	63.3 \pm 3.0\dagger	18.6 \pm 2.4
Color vs Animal (5)	47.0 \pm 3.1\dagger	9.5 \pm 1.8	Adjective vs Verb (3)	57.3 \pm 3.1	56.1 \pm 3.1
Adjective vs Verb (5)	52.3 \pm 3.1\dagger	39.4 \pm 3.0	Verb vs Adjective (3)	70.5 \pm 2.8\dagger	33.6 \pm 2.9
Verb vs Adjective (5)	61.3 \pm 3.0\dagger	20.8 \pm 2.5	Concept vs Object (3)	56.8 \pm 3.1\dagger	35.9 \pm 3.0
Concept vs Object (5)	45.7 \pm 3.1\dagger	18.3 \pm 2.4	Object vs Concept (3)	73.2 \pm 2.7\dagger	58.4 \pm 3.1
Object vs Concept (5)	68.9 \pm 2.9\dagger	51.9 \pm 3.1	SQuAD Validation	17.7 \pm 1.1\dagger	8.5 \pm 0.8

773
774
775
776
777
778
779Table A2: Performance comparison on **abstractive** in-context learning tasks (5-Shot) for Thresholded-HAPAX. Values show accuracy \pm 95% CI margin. Bold indicates higher performance; \dagger denotes statistical significance ($p < 0.05$) according to McNemar’s test.

Task	Vanilla (%)	HAPAX THRESH (%)	Task	Vanilla (%)	HAPAX THRESH (%)
Capitalize First Letter	37.3 \pm 3.3\dagger	26.9 \pm 3.0	Singular-Plural	67.8 \pm 6.4\dagger	39.5 \pm 6.7
Country-Capital	31.5 \pm 6.5	32.0 \pm 6.5	AG News	32.9 \pm 1.3\dagger	1.1 \pm 0.3
Lowercase First Letter	72.2 \pm 3.1\dagger	48.2 \pm 3.4	Synonym	1.6 \pm 0.5	4.4 \pm 0.7\dagger
CommonsenseQA	19.8 \pm 1.1\dagger	0.0 \pm 0.0	Capitalize Second Letter	11.7 \pm 2.2\dagger	4.1 \pm 1.4
Landmark-Country	32.9 \pm 3.2\dagger	21.8 \pm 2.8	Product-Company	20.1 \pm 3.4\dagger	9.0 \pm 2.5
Capitalize Last Letter	9.7 \pm 2.0\dagger	4.1 \pm 1.4	Antonym	0.8 \pm 0.3	7.8 \pm 1.1\dagger
Lowercase Last Letter	6.8 \pm 1.7\dagger	3.2 \pm 1.2	Word Length	7.9 \pm 1.8\dagger	1.2 \pm 0.8
Next Item	12.0 \pm 4.2	10.2 \pm 4.0	National Parks	20.4 \pm 3.7	19.7 \pm 3.7
Country-Currency	29.9 \pm 6.4\dagger	12.2 \pm 4.6	Capitalize (Full Word)	76.9 \pm 2.9\dagger	46.1 \pm 3.4
Present-Past	54.3 \pm 5.7\dagger	40.3 \pm 5.6	Previous Item	5.8 \pm 3.0\dagger	1.8 \pm 1.7
Park-Country	12.4 \pm 2.4	15.2 \pm 2.6\dagger	Next Capital Letter	6.4 \pm 1.7	4.5 \pm 1.4
Sentiment	65.3 \pm 2.7\dagger	0.0 \pm 0.1			

790
791
792Table A3: Performance comparison on **extractive** in-context learning tasks (5-Shot) for Thresholded-HAPAX. Values show accuracy \pm 95% CI margin. Bold indicates higher performance; \dagger denotes statistical significance ($p < 0.05$) according to McNemar’s test.

Task	Vanilla (%)	HAPAX THRESH (%)	Task	Vanilla (%)	HAPAX THRESH (%)
Choose First of 3	95.0 \pm 1.4\dagger	30.7 \pm 2.9	Choose First of 5	95.9 \pm 1.2\dagger	27.0 \pm 2.8
Choose Middle of 3	40.2 \pm 3.0\dagger	21.5 \pm 2.5	Choose Middle of 5	26.7 \pm 2.7\dagger	7.4 \pm 1.6
Choose Last of 3	66.1 \pm 2.9\dagger	53.6 \pm 3.1	Choose Last of 5	52.2 \pm 3.1	58.7 \pm 3.1\dagger
Alphabetically First (3)	34.4 \pm 2.9\dagger	28.1 \pm 2.8	Alphabetically First (5)	19.9 \pm 2.5\dagger	14.5 \pm 2.2
Alphabetically Last (3)	32.3 \pm 2.9\dagger	24.5 \pm 2.7	Alphabetically Last (5)	18.5 \pm 2.4\dagger	13.0 \pm 2.1
CoNLL: Person	73.2 \pm 1.5\dagger	25.8 \pm 1.4	CoNLL: Location	59.4 \pm 1.4\dagger	17.5 \pm 1.1
CoNLL: Organization	57.8 \pm 1.6\dagger	21.2 \pm 1.3	Animal vs Object (3)	48.4 \pm 3.1\dagger	27.3 \pm 2.8
Animal vs Object (5)	36.3 \pm 3.0\dagger	16.3 \pm 2.3	Fruit vs Animal (3)	45.1 \pm 3.1\dagger	21.5 \pm 2.5
Fruit vs Animal (5)	28.1 \pm 2.8\dagger	9.9 \pm 1.9	Color vs Animal (3)	63.3 \pm 3.0\dagger	18.6 \pm 2.4
Color vs Animal (5)	47.0 \pm 3.1\dagger	9.5 \pm 1.8	Adjective vs Verb (3)	57.3 \pm 3.1	56.1 \pm 3.1
Adjective vs Verb (5)	52.3 \pm 3.1\dagger	39.4 \pm 3.0	Verb vs Adjective (3)	70.5 \pm 2.8\dagger	33.6 \pm 2.9
Verb vs Adjective (5)	61.3 \pm 3.0\dagger	20.8 \pm 2.5	Concept vs Object (3)	56.8 \pm 3.1\dagger	35.9 \pm 3.0
Concept vs Object (5)	45.7 \pm 3.1\dagger	18.3 \pm 2.4	Object vs Concept (3)	73.2 \pm 2.7\dagger	58.4 \pm 3.1
Object vs Concept (5)	68.9 \pm 2.9\dagger	51.9 \pm 3.1	SQuAD Validation	17.7 \pm 1.1\dagger	8.5 \pm 0.8

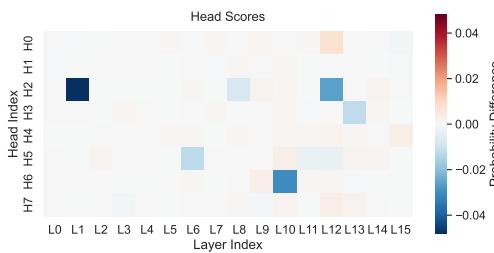
809

810
 811 Table A4: Performance comparison on **abstractive** in-context learning tasks (5-Shot) by ensuring
 812 that the target answer for each test instance is not included in the 5-shot examples. Values show
 813 accuracy \pm 95% CI margin. Bold indicates higher performance; \dagger denotes statistical significance
 814 ($p < 0.05$) according to McNemar's test.

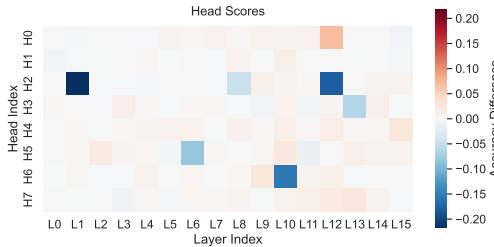
Task	Vanilla (%)	HAPAX (%)	Task	Vanilla (%)	HAPAX (%)
AG News	0.3 \pm 0.3	7.8 \pm 1.7\dagger	Antonym	0.8 \pm 0.6	2.5 \pm 1.0\dagger
Capitalize Second Letter	0.5 \pm 0.5	3.2 \pm 1.2\dagger	CommonsenseQA	12.0 \pm 2.0	13.9 \pm 2.1
Country-Capital	30.7 \pm 6.6	42.3 \pm 7.0\dagger	Capitalize (Full Word)	79.1 \pm 2.8\dagger	62.2 \pm 3.3
Capitalize First Letter	36.4 \pm 3.3	73.6 \pm 3.0\dagger	Capitalize Last Letter	2.9 \pm 1.2	6.6 \pm 1.7\dagger
Country-Currency	4.3 \pm 4.1	4.3 \pm 4.1	Lowercase First Letter	73.3 \pm 3.0	76.2 \pm 2.9
National Parks	15.3 \pm 3.3	22.6 \pm 3.9\dagger	Next Capital Letter	3.4 \pm 1.3	8.6 \pm 1.9\dagger
Landmark-Country	30.4 \pm 3.1	38.4 \pm 3.3\dagger	Lowercase Last Letter	3.8 \pm 1.3	10.0 \pm 2.1\dagger
Next Item	12.0 \pm 4.2	28.0 \pm 5.9\dagger	Park-Country	10.7 \pm 2.2	16.7 \pm 2.7\dagger
Present-Past	54.6 \pm 5.7	78.8 \pm 4.7\dagger	Previous Item	5.3 \pm 2.9	7.1 \pm 3.4
Product-Company	15.9 \pm 3.1	24.9 \pm 3.7\dagger	Sentiment	2.8 \pm 1.0	35.4 \pm 3.0\dagger
Singular-Plural	62.0 \pm 6.6	77.1 \pm 5.8\dagger	Synonym	1.6 \pm 0.8	2.0 \pm 0.9
Word Length	7.7 \pm 1.8	14.9 \pm 2.4\dagger			

826
 827 Table A5: Performance comparison on **abstractive** in-context learning tasks (5-Shot) for
 828 THRESHOLDED-HAPAX by ensuring that the target answer for each test instance is not included
 829 in the 5-shot examples. Values show accuracy \pm 95% CI margin. Bold indicates higher performance;
 830 \dagger denotes statistical significance ($p < 0.05$) according to McNemar's test.

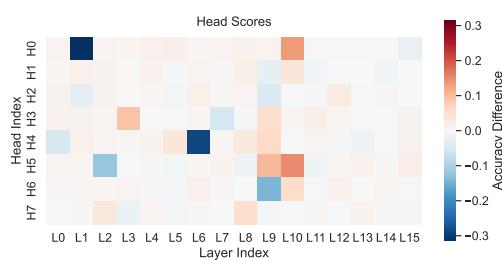
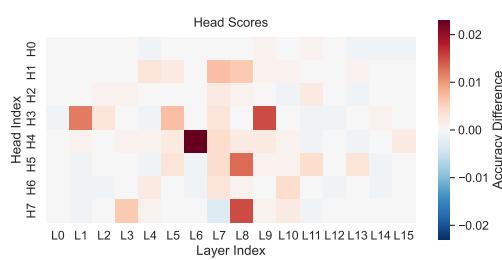
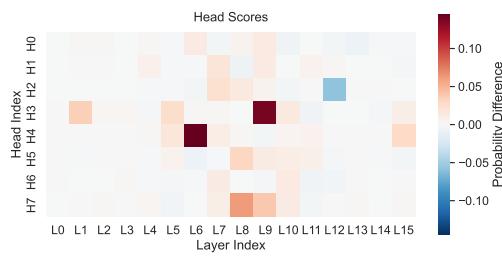
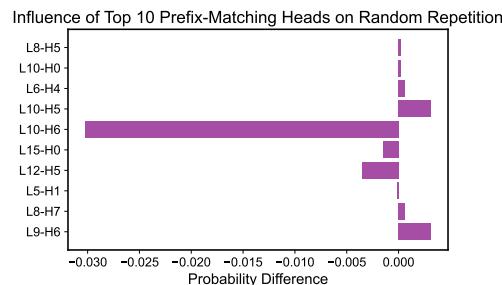
Task	Vanilla (%)	HAPAX THRESH (%)	Task	Vanilla (%)	HAPAX THRESH (%)
AG News	0.3 \pm 0.3	3.7 \pm 1.2\dagger	Antonym	0.8 \pm 0.6	7.0 \pm 1.6\dagger
Capitalize Second Letter	0.5 \pm 0.5	7.6 \pm 1.9\dagger	CommonsenseQA	12.0 \pm 2.0\dagger	7.3 \pm 1.6
Country-Capital	30.7 \pm 6.6\dagger	29.6 \pm 6.5	Capitalize (Full Word)	79.1 \pm 2.8\dagger	52.2 \pm 3.4
Capitalize First Letter	36.4 \pm 3.3\dagger	31.2 \pm 3.2	Capitalize Last Letter	2.9 \pm 1.2	8.2 \pm 1.9\dagger
Country-Currency	4.3 \pm 4.1\dagger	3.2 \pm 3.4	Lowercase First Letter	73.3 \pm 3.0\dagger	69.2 \pm 3.2
National Parks	15.3 \pm 3.3	25.3 \pm 4.0\dagger	Next Capital Letter	3.4 \pm 1.3	6.3 \pm 1.7\dagger
Landmark-Country	30.4 \pm 3.1\dagger	26.1 \pm 3.0	Lowercase Last Letter	3.8 \pm 1.3	5.2 \pm 1.5\dagger
Next Item	12.0 \pm 4.2	12.0 \pm 4.2	Park-Country	10.7 \pm 2.2	16.1 \pm 2.7\dagger
Present-Past	54.6 \pm 5.7\dagger	41.6 \pm 5.6	Previous Item	5.3 \pm 2.9\dagger	1.8 \pm 1.7
Product-Company	15.9 \pm 3.1	20.7 \pm 3.5\dagger	Sentiment	2.8 \pm 1.0\dagger	0.0 \pm 0.1
Singular-Plural	62.0 \pm 6.6\dagger	34.6 \pm 6.5	Synonym	1.6 \pm 0.8	3.8 \pm 1.2\dagger
Word Length	7.7 \pm 1.8\dagger	2.0 \pm 1.0			



843
 844 Figure A1: Probability difference values of random repetition task for Thresholded-HAPAX.
 845



854
 855 Figure A2: Accuracy difference values of random repetition task for Thresholded-HAPAX.
 856

874
875
876
877
878
879
880
881
882
883
884
885
886887
888
889
890
891
892
893
894
895
896
897
898
899900
901
902
903
904
905
906
907
908
909
910
911
912
913
914915
916
917

918
919
920
921
922
923
924
925
926
927
928
929
930
931

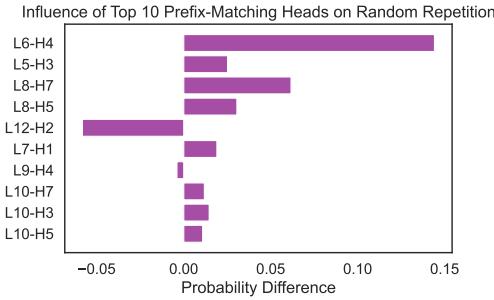
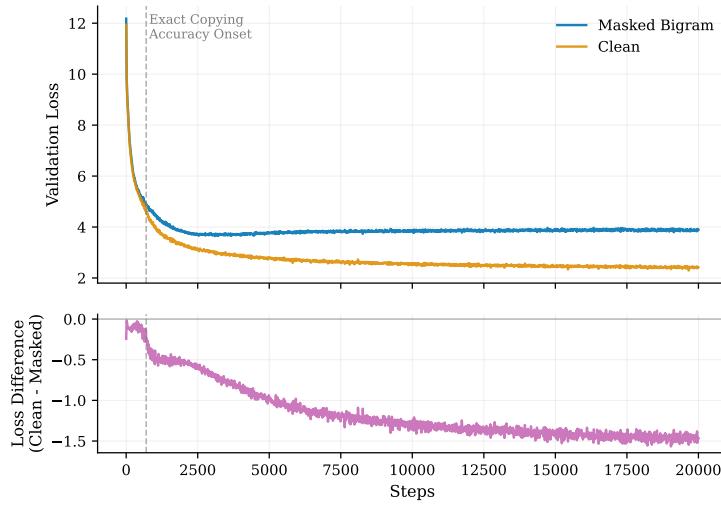


Figure A7: Probability Difference for random repetition task of the Top 10 Prefix-Matching heads for vanilla model.

932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950



951
952
953
Figure A8: Validation Loss of Vanilla and Hapax model. Despite having increased loss value, as
mentioned in the main paper, the validation loss also is mostly affected by the prediction of exact
copying instances.

954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971

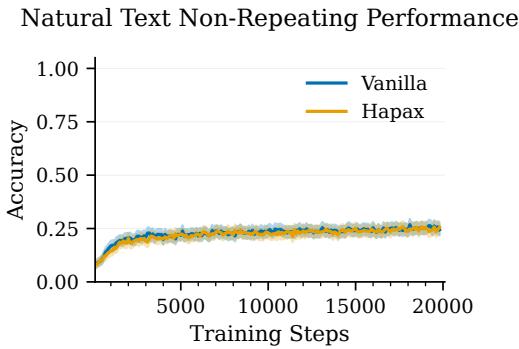
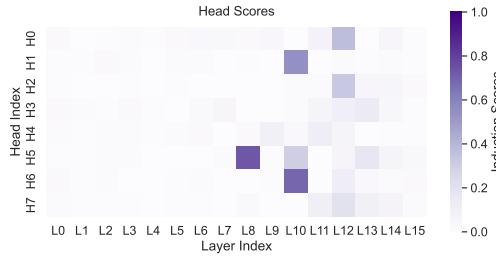
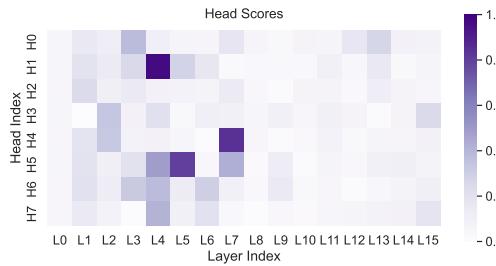


Figure A9: The non-repeating natural text repetition accuracy. Instead of giving a repeated sequence $r_1 r_2 \dots r_s r_1 r_2 \dots r_{s-1}$ we only give $r_1 r_2 \dots r_{s-1}$. This shows that the initial increase seen in the natural text repetition also occurs for non-repeating samples.

981 Figure A10: Prefix Matching Scores for Thresholded-HAPAX.
982992 Figure A11: Previous Token Head of vanilla model at step 5000. L4 is the first layer that has a
993 strong previous token head, which is L4 H1.
994995
996 A.3 ATTENTION PATTERN SCORES
997998 B COSINE SIMILARITY THRESHOLD SELECTION
999

1000 The main purpose of Thresholded-HAPAX is to create a stricter training regime for suppression of
1001 inductive copying. Our Thresholded-HAPAX experiments are run to observe how well this masking
1002 reduces copying signal compared to both the regular model and the regular HAPAX model. In Figure
1003 3a, the regular HAPAX model has highly reduced copying accuracy compared to the regular model
1004 but still at a non-trivial level. Our hypothesis is that even if we mask exact verbatim copying signals,
1005 there might still be token representations with high representational similarity, high enough to still
1006 provide a weak signal for verbatim copying. To test this, we select the cosine similarity threshold by
1007 using the embedding space of the last step of the vanilla model as a proxy for finding similar tokens.
1008 We create a cosine similarity matrix and analyze the cosine similarities of the closest neighbors
1009 and their edit distances. After doing this, we select the threshold of 0.3, which corresponds to the
1010 top-4 closest neighbors on average (Figure A12) and an average edit distance of 3.65 (Figure A13).
1011 Tokens representing different strings may still have highly similar embeddings, so our thresholding
1012 deliberately overestimates these hypothesized similar tokens, which makes us more confident that
1013 we are removing copying signal. Given our limited compute budget, we adopt this strict threshold
1014 to ensure that the trained model consistently masks the majority of similar tokens. Although this
1015 eventually causes harm for abstractive tasks when compared to the regular HAPAX model, we ob-
1016 serve consistent gains for translation, suggesting that the model keeps capabilities for such instances
1017 despite the heavy masking. This control over copying also cannot solely be explained by under-
1018 training, because the Thresholded-HAPAX model is near zero at copying while still achieving 29.6%
1019 accuracy on non-trivial tasks like Country-Capital. Therefore, even if it performs worse than the
1020 vanilla model, it serves as a proof of concept that the model can achieve non-trivial capabilities in
1021 abstractive tasks despite having even more reduced copying and prefix-matching scores.

1022 To further justify our use of input embedding cosine similarity (instead of edit distance), we analyze
1023 the relationship between string-level edit distance and embedding cosine similarity for the top-50
1024 nearest neighbors in the vanilla model’s embedding space (Figure A15). With a cosine similarity
1025 threshold of 0.3, 7.24% of the token pairs are treated as equal, and only 1% of these pairs have an
edit distance higher than 3. In contrast, if we instead threshold by edit distance at 3, 34.8% of the
pairs are treated as equal, and 28.56% of these have a cosine similarity below 0.3. Moreover, an

1026 edit distance threshold of 3 still fails to capture 1% of the pairs whose cosine similarity is above 0.3.
 1027 Therefore, edit distance treats many more pairs as equal but still does not capture some high-cosine-
 1028 similarity token pairs, which we hypothesize are more closely related to copying signal. Since
 1029 attention heads attend by hidden state similarity, cosine similarity is closer to what an attention head
 1030 would consider similar. We use cosine similarity as our base and use the average edit distance at
 1031 different cosine similarity thresholds to guide us, so that we make sure we are removing tokens
 1032 above a certain cosine similarity while also understanding how close they are on average to their
 1033 represented tokens.

1034

1035

1036

1037

1038

1039

1040

1041

1042

1043

1044

1045

1046

1047

1048

1049

1050

1051

1052

1053

1054

1055

1056

1057

1058

1059

1060

1061

1062

1063

1064

1065

1066

1067

1068

1069

1070

1071

1072

1073

1074

1075

1076

1077

1078

1079

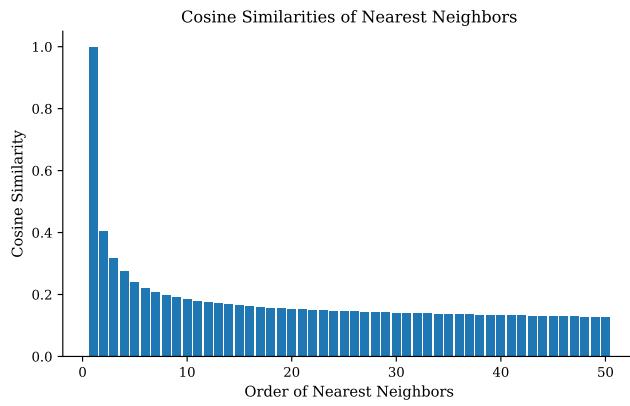


Figure A12: Cosine similarities of nearest N neighbors for clean model.

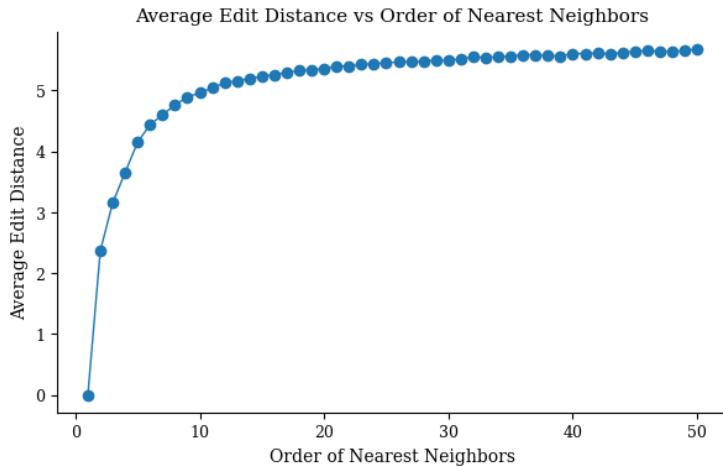


Figure A13: Average edit distance of nearest neighbors for vanilla model.

C HAPAX FLUENCY EVALUATION

We evaluate the effects of HAPAX training on the model’s ability to generate fluent natural text and compare the results to its vanilla counterpart as a baseline. Inspired by the Fluency score from Axbench Wu et al. (2025), we use an automated LLM-based approach to assess the fluency of sentence completions generated by each model, with discrete ratings of 0, 1, and 2 for fluent language. Sequence continuations, with a maximum length of 128 tokens, are generated based on a diverse seed of 20 tokens, sampled from the RedPajama dataset Weber et al. (2024).

We find in Figure A16 that our HAPAX trained model generates somewhat fluent text 70% more times than the vanilla model. The latter, performing poorly, generates non-fluent language 84% of the time. Looking at some qualitative examples of generation in Figure A17, we observe that the

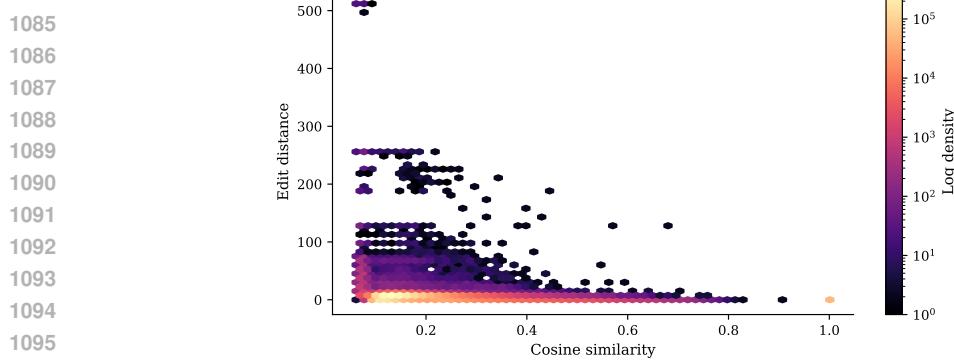
1080

1081

1082

1083

1084 Density of Token Pairs by Edit Distance and Cosine Similarity



1097 Figure A14: Low edit-distance pairs appear across a broad range of cosine similarities, suggesting
 1098 that edit distance over-treats tokens as equivalent. The analysis is done on the same pairs analyzed
 1099 in A13.

1100

1101

1102

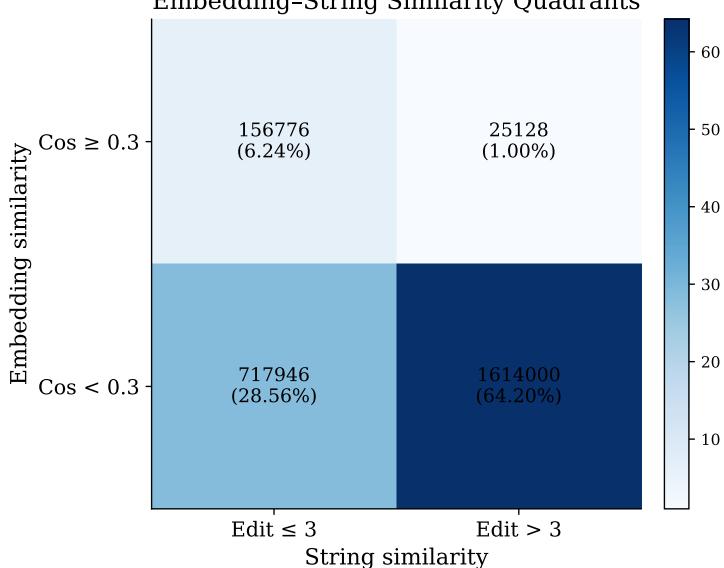
1103

1104

1105

1106

1107 Embedding-String Similarity Quadrants



1127 Figure A15: Relationship between string-level similarity (edit distance) and embedding similarity
 1128 for the top-50 nearest neighbors. Many token pairs with low edit distance also have low embedding
 1129 similarity (28.56%). However, 1% of the token pairs with cosine similarity above 0.3 are still not
 1130 captured. Since attention heads attend by hidden state similarity, we want to make sure we are
 1131 removing such similar tokens, which motivates our use of cosine similarity.

1132

1133

vanilla model defaults to repeating itself rather early, creating incoherent language, which explains its negative results. On the other hand, the HAPAX model is able to produce chained narratives with much more fluid and natural structure.

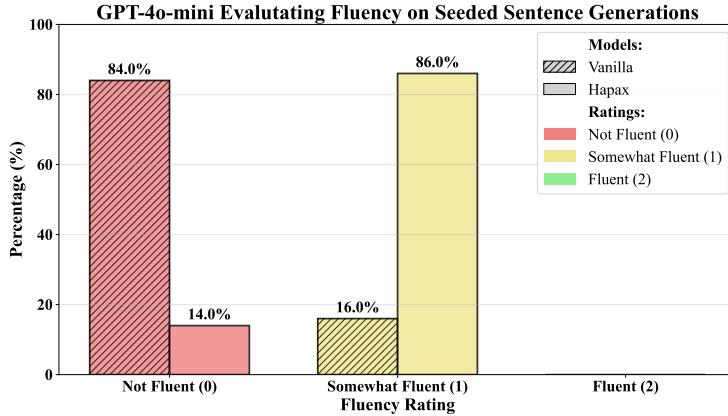


Figure A16: GPT-4o-mini as a judge rates 100 samples of natural text generated by each model based on fluency.

D EXPERIMENTAL DETAILS

D.1 TRANSLATION TASKS

To perform translation evaluation, we constructed a multilingual translation dataset using the Open Multilingual Wordnet (OMW) 1.4 corpus via NLTK (Bird & Loper, 2004). To select languages, we used W3Techs¹ and chose languages that are both present in OMW 1.4 and have above 1% usage on the internet according to W3Techs. Since W3Techs data update dynamically, our selection was based on June 2025 data. Our choices resulted in 981 parallel concepts to be used for our tasks. We use accuracy as our evaluation metric. For each synset, we use only the first lemma from the source language to predict the corresponding English translation. All evaluations are 5-shot, and we translate from each source language into English to ensure consistent token counts across all tasks. An example prompt is shown in Figure A18.

E EXPLORED TRAINING VARIANTS

Before finalizing the HAPAX training protocol, we explored several variants to understand the impact of inductive copying on learning abstractive ICL capabilities. The methods proposed here focus mainly on the first part of the induction circuit, namely the prefix-matching behavior. The motivation behind these methods is to eliminate the initial circuit component, which we expected would remove inductive copying behavior completely. We summarize these variants and discuss why they fail and are therefore not suitable for our purposes.

E.1 LOSS PENALTY BASED INTERVENTIONS FOR PREFIX-MATCHING SCORE

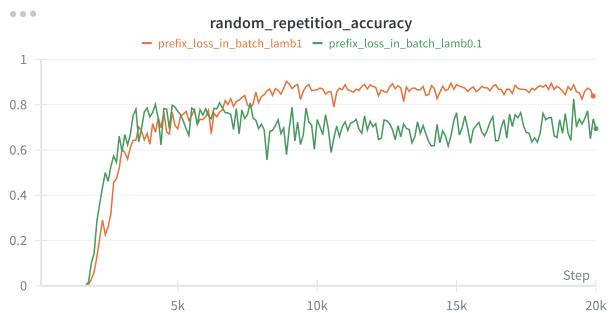
To analyze the impact of inductive copying on the development of abstractive ICL capabilities, we included a loss function that penalizes the prefix-matching attention pattern commonly observed in induction heads. Given our loss function \mathcal{L} , we add the prefix-matching score across all heads to our loss function:

$$\mathcal{L}_{\text{pm}} = \lambda \frac{1}{LH} \sum_{l=1}^L \sum_{h=1}^H \text{PrefixMatching}(l, h), \quad (5)$$

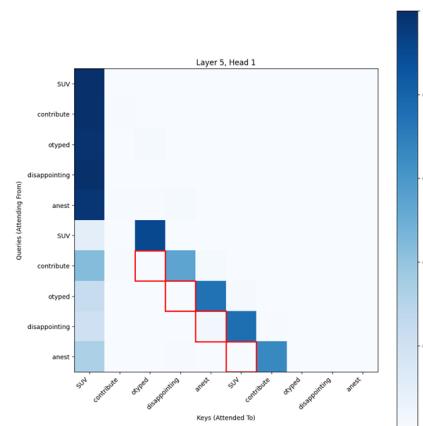
¹https://w3techs.com/technologies/overview/content_language

1242 where λ is our hyperparameter for fine-tuning. We propose two different versions. In the first one,
 1243 we compute the prefix-matching score on the synthetically generated random repetition sequence.
 1244 The random sequences are given in separate forward passes along with regular training. In the
 1245 second version, instead of using synthetic data, we compute the prefix-matching score on the training
 1246 data itself. We use the same positions as we use to mask the loss in Section 3.1.

1247 The initial approach overfits on the synthetic data and consequently, the prefix-matching pattern
 1248 and copying is intact for natural language. The second approach reduces prefix-matching score
 1249 for both synthetic and natural settings. However, although we observe that the loss function can
 1250 control the prefix-matching score, the model can still do random copying. We give random repetition
 1251 accuracies for two different lambda values in Figure A19 for the non-synthetic variation. When we
 1252 analyze individual attention patterns on the random repetition task, we find that the model learns to
 1253 copy from two-tokens ahead, instead of displaying the regular prefix-matching attention pattern. As
 1254 an example, in Figure A20, red squares indicate the regular prefix-matching positions. However,
 1255 when we penalize the attention pattern, the model can still read the information from the 1-offset
 1256 positions. Therefore, attention-pattern based training interventions are ill-suited for our purposes.
 1257 This observation also motivate us to focus on *incentives*, as we did for HAPAX, rather than directly
 1258 optimizing for a specific behavior, since the model’s solution space tends to discover alternative
 1259 ways to achieve the objective, similar to reward hacking.



1272 Figure A19: Random repetition accuracy for models trained with prefix-matching loss. Two lambda
 1273 values that reduces the prefix-matching score are still able to learn random repetition.



1291 Figure A20: Example attention pattern of a model trained with a loss function that penalizes prefix-
 1292 matching scores. Red squares indicate prefix-matching positions. Although the model does not
 1293 display the typical attention pattern for induction, it learns to read the copying information from
 1294 1-offset positions.

1296
1297

E.2 REINITIALIZING INDUCTION HEADS

1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310

To analyze whether weight reinitialization would allow us to suppress inductive copying, we have created two different training protocols. First, we reinitialized any attention head whose prefix-matching score was above a threshold τ , which is 0.3 in our case. Secondly, in addition to reinitializing the weights, we have reinitialized the optimizer states for the corresponding attention heads that go above the threshold. For the first case, we have observed that the model can continue its training but the reinitialization creates more induction heads as compared to what it would learn in the vanilla version (Figure A22). We hypothesize that this might be due to the fact that ADAM optimizer builds up enough momentum which causes different heads to converge into induction heads as the training goes on. When we also reinitialize the corresponding optimizer states, the model training procedure gets stuck (Figure A21, A23) and causes every attention head to have an increased prefix-matching score. Therefore, when we keep the same data distribution, the model tends to converge on similar solutions for copying behavior, which makes the presented methods unsuitable for our purposes.

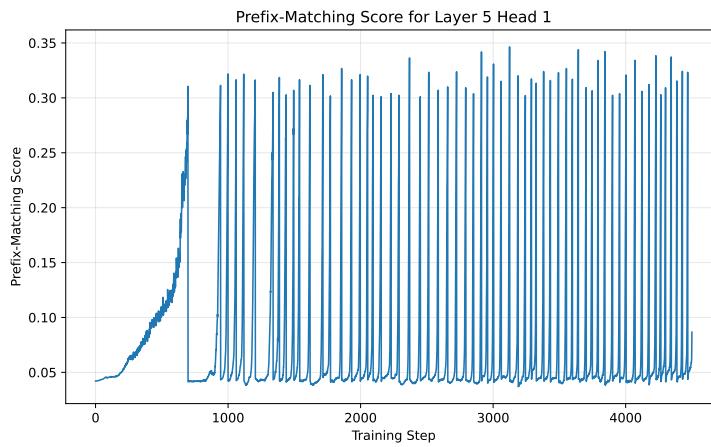
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325

Figure A21: When the optimizer state for induction heads is reinitialized in combination with the weights, the model gets stuck and tries to relearn induction heads. We plot prefix-matching scores before we reinitialize, which is why certain data points exceed 0.3.

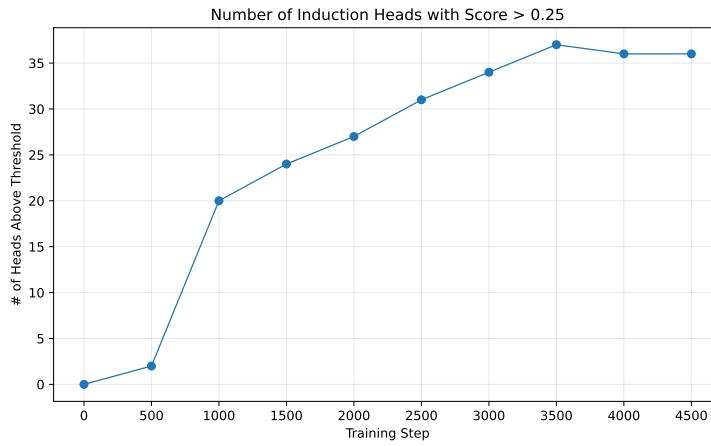
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346

Figure A22: Number of induction heads that exceed the 0.25 threshold for the reinitialized model (160M). Since we reinitialize heads that exceed 0.3, we plot heads that exceed 0.25. The reinitialization causes the model to create three times more induction heads as compared to its vanilla version.

1350

1351

1352

1353

1354

1355

1356

1357

1358

1359

1360

1361

1362

1363

1364

1365

1366

1367

1368

1369

1370

1371

1372

1373

1374

1375

1376

1377

1378

1379

1380

1381

1382

1383

1384

1385

1386

1387

1388

1389

1390

1391

1392

1393

1394

1395

1396

1397

1398

1399

1400

1401

1402

1403

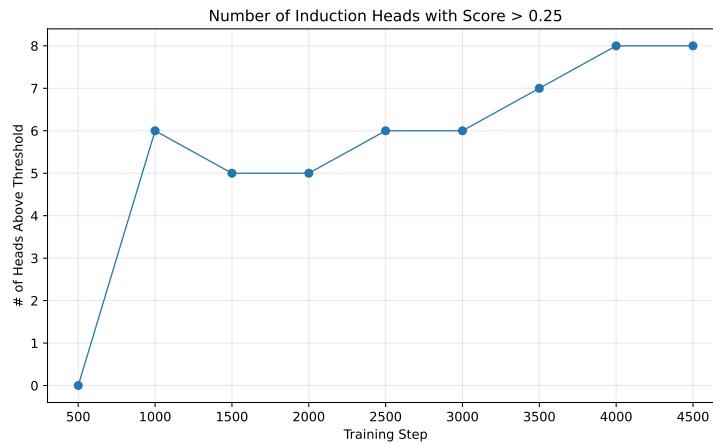


Figure A23: Number of attention heads that exceed 0.25 threshold for a vanilla model (160M).

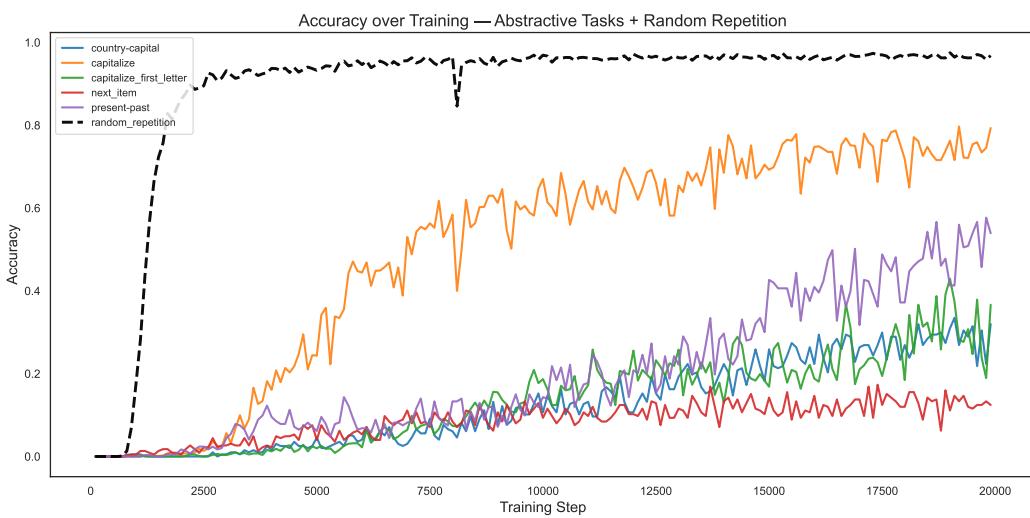


Figure A24: Accuracies of sample tasks over training for the Vanilla model. Random repetition accuracy is the first task to improve, and its rise coincides with the training step where the loss drops sharply (A8). This motivates examining whether the decrease in loss and the increase in inductive copying translate into meaningful gains on abstractive tasks, which according to our evidence are not tightly coupled.

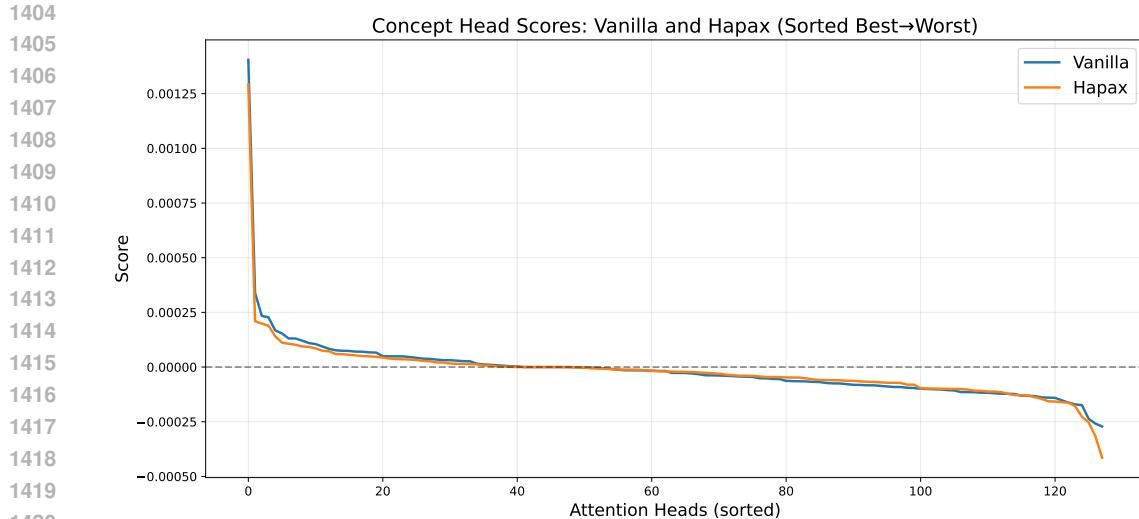


Figure A25: Using the concept head scores from Feucht et al. (2025), we display the scores of the models from best to worst. The trends are similar for both the Vanilla and HAPAX models, indicating that suppressing inductive copying does not noticeably affect the causal concept head metrics.

F USAGE OF LLMs

In accordance with the ICLR 2026 author guidelines, we made limited use of LLM-based services such as Claude and Grammarly as assistive tools for correcting grammar and refine writing.

G CHOICE OF STATISTICAL TESTING

For downstream tasks where evaluation reduces to accuracy, each example yields a paired binary outcome across models (correct vs. incorrect). McNemar’s test is appropriate for this paired binary setting, as it assesses whether disagreements between the two models on the same examples are symmetric. Prior work has shown that McNemar’s test is well-calibrated for classifier comparison in such conditions Dietterich (1998). Accordingly, we use McNemar’s test for our main accuracy-based comparisons.