LEARNING ON LLM OUTPUT SIGNATURES FOR GRAY BOX LLM BEHAVIOR ANALYSIS

Guy Bar-Shalom*
Technion
guybs99@gmail.com

Fabrizio Frasca*
Technion
fabriziof@campus.technion.ac.il

Derek Lim
MIT CSAIL

Yoav Gelberg
Technion

Yftah Ziser
Nvidia

Ran El-Yaniv
Technion, Nvidia

Gal Chechik
Bar-Ilan University, Nvidia

Haggai Maron
Technion, Nvidia

ABSTRACT

Large Language Models (LLMs) have achieved widespread adoption, yet our understanding of their behavior remains limited, particularly in detecting data contamination and hallucinations. While recently proposed probing techniques provide insights through activation analysis, they require "white-box" access to model internals, often unavailable. Current "gray-box" approaches typically analyze only the probability of the actual tokens in the sequence with simple task-specific heuristics. Importantly, these methods overlook the rich information contained in the full token distribution at each processing step. To address these limitations, we propose that gray-box analysis should leverage the complete observable output of LLMs, consisting of both the previously used token probabilities as well as the complete token distribution sequences - a unified data type we term LOS (LLM Output Signature). To this end, we develop a transformer-based approach to process LOS that theoretically guarantees approximation of existing techniques while enabling more nuanced analysis. Our approach achieves superior performance on hallucination and data contamination detection in gray-box settings, significantly outperforming existing baselines. Furthermore, it demonstrates strong transfer capabilities across datasets and LLMs, suggesting that LOS captures fundamental patterns in LLM behavior. Our code is available at: https://github.com/BarSGuy/LLM-Output-Signatures-Network.

1 INTRODUCTION

Large Language Models (LLMs) have demonstrated remarkable capabilities across diverse applications, yet their internal mechanisms remain poorly understood. This gap in understanding is particularly relevant in critical tasks like Hallucination Detection (HD) Tonmoy et al. (2024); Liu et al. (2021); Huang et al. (2023a); Ji et al. (2023); Rawte et al. (2023) and Data Contamination Detection (DCD) Brown et al. (2020); Shi et al. (2023); Zhang et al. (2024): in these contexts, determining whether an LLM is fabricating information or has been exposed to specific training data is crucial for deployment safety and reliability.

Previous work on LLM analysis has relied heavily on probing techniques that require restrictive white-box access to model internals (Belinkov, 2022; Orgad et al., 2024; Hewitt & Manning, 2019; Hewitt & Liang, 2019; Rateike et al., 2023). Gray-box methods relax these assumptions by operating *only* on LLM outputs. Existing gray-box approaches typically analyze just the sequence of probabilities assigned to tokens that appear in the relevant input or output token sequence – a vector we term Actual Token Probabilities (ATP) (Guerreiro et al., 2022; Kadavath et al., 2022; Varshney et al., 2023; Huang et al., 2023b). However, these methods, often based on heuristics, overlook the information contained in the complete Token Distribution Sequence (TDS) – a matrix holding the next-token probability distribution at each generation step, see Figure 1. This limitation can mask crucial differences in model behavior even at the level of a single time step. E.g., consider a model

*Equal contribution

Figure 1: A visualization of the LLM Output Signature (LOS). Left: The LLM processes the input "What does the cat chase?" and generates the output "A big mouse". Right: The corresponding query/response Token Distribution Sequence (TDS) – full next-token distributions over the sequence – and Actual Token Probabilities (ATP), which represent the probabilities of tokens in the actual sequence (for the query) and the sampled token probabilities (for the response). We propose learning directly from this unified LOS representation to analyze LLM behavior.

generating a token with probability 0.5 in two scenarios: in one case, the remaining next-token probability mass is concentrated on a single alternative $(0.5, 0.5, 0, ..., 0)$, while in the other it is spread across many tokens: $(0.5, 0.01, ..., 0.01)$. These distributions suggest very different levels of model uncertainty, yet ATP-based approaches would treat them identically. Similarly, an ATP value of 0.1 at a certain time step could indicate either high uncertainty (if it is the highest probability in a diffused distribution) or strong evidence against the token (if it is a low-ranking probability in a peaked distribution). A recent promising approach Zhang et al. (2024) used some TDS information through heuristics, but a principled framework to utilize this data lacks.

Our approach. We argue that a successful gray-box approach should leverage both ATP and TDS, together forming what we term the LLM Output Signature (LOS) (Figure 1) – the complete observable representation of LLM behavior in the gray-box setup. Instead of relying on heuristics, we treat LOS as a sequential, high-dimensional and structured data modality on which we apply principled deep learning techniques. We propose LOS-NET, a lightweight transformer encoder[1] that operates on an effective encoding of ATP, TDS, and their interactions. From a theoretical perspective, we show that LOS-NET can provably approximate a broad class of functions applied to the LOS of any LLM, subsuming many recent approaches (Guerreiro et al., 2022; Kadavath et al., 2022; Varshney et al., 2023; Huang et al., 2023b; Shi et al., 2023; Zhang et al., 2024). Through a comprehensive empirical study on the tasks of DCD and HD, we demonstrate that the information gap between using the complete LOS and relying solely on ATP is substantial. We improve over all considered baselines across both tasks – often by a significant margin. Notably, LOS-NET exhibits promising dataset-level transfer and strong cross-model generalization, suggesting it can capture universal patterns in LLM behavior. Importantly, the cross-model transfer abilities of LOS-NET suggest its viable application to impactful real-world tasks such as copyright-infringement detection over closed-source LLMs, as indicated by our results on the BookMIA benchmark (Shi et al., 2023).

Contributions are summarized as follows: (1) we introduce LOS as a suitable representation for analyzing LLM behavior, (2) we develop an effective learning framework for the LOS data modality, (3) we show this unifies and generalizes previous approaches, (4) we demonstrate it achieves superior performance across models, datasets, and tasks, and (5) exhibits strong empirical evidence for cross-model generalization and promising cross-dataset transfer abilities. The proposed LOS-NET proves effective for both HD and DCD, and its flexibility suggests broader potential for similar tasks while paving the way for foundational approaches to modeling LLM behaviors.

2 RELATED WORK

We review background and related work on DCD and HD, focusing on studies leveraging logits or output probabilities. Given the breadth of research, we highlight the most relevant works for our setup and refer interested readers to Appendix D for further details on these tasks.

Data Contamination Detection. Early methods leveraged model loss Yeom et al. (2018); Carlini et al. (2019) for DCD, assuming that models overfit their training data. Later refinements introduced reference models—independent LLMs trained on disjoint datasets from a similar distribution—comparing their scores with the target model Carlini et al. (2021; 2022). However, this approach depends on the availability of a well-matched reference model (similar in its architecture), which is often impractical. Recently, Shi et al. (2023) introduced Min-K%, which flags an input as

[1] Around 1M parameters.

contaminated if the log probability of its bottom K tokens exceeds a predefined threshold. Building on this approach, Zhang et al. (2024) proposed Min-K%++, which refines contamination detection by calibrating the next-token log-likelihood using the mean and standard deviation of log-likelihoods across all candidate tokens in the vocabulary.

Hallucination Detection. Hallucination detection has been studied as a means of enabling selective intervention, allowing LLMs to prevent fabricated outputs only when necessary Snyder et al. (2024); Yin et al. (2024); Valentin et al. (2024). Recently, Orgad et al. (2024) showed that training a classifier on top of LLMs' hidden states (intermediate activations) is highly effective for hallucination detection. However, this method operates under the white-box assumption, requiring full access to the model's internal components. In contrast, our paper explores a more constrained (gray-box) setting.

Output probability-Based Analysis. Previous works showed that using log probabilities or raw logits as decision thresholds can be effective for various tasks, including HD in LLMs Guerreiro et al. (2022); Varshney et al. (2023), correctness self-evaluation Kadavath et al. (2022), uncertainty estimation Huang et al. (2023b), and zero shot learning Atzmon & Chechik (2019). However, these approaches often rely on naive handcrafted thresholding. Other approaches feed probabilities or logits into classifiers to get a more refined signal. Mosca et al. (2022) computes logit differences for texts with and without a given word, training a classifier to detect adversarial attacks. Wu et al. (2023) introduced LLMDet, which quantifies perplexity scores across models by analyzing next-token probabilities for selected n-grams, feeding these into a classifier to detect machine-generated content. Similarly, Verma et al. (2024) presented Ghostbuster, which extracts token probabilities using simpler models and trains a linear classifier for the same aforementioned task. Both rely on linear classifiers and overlook the LLM's TDS, limiting contextual understanding. In contrast, our method fully leverages textual context via the LOS for a more nuanced analysis.

3 Learning on LLM Output Signatures

3.1 Notation and Problem Formulation

Let f denote a pretrained LLM, and $\vec{s}$ refer to a text input to f consisting of n tokens. When queried with $\vec{s}$, the LLM f produces outputs $\mathbf{X}_s = f(\vec{s})$, i.e., a matrix in $\mathbb{R}^{n \times V}$ of next-token probabilities for each token in $\vec{s}$, where V is the size of the token vocabulary. We define the LLM response to be $\vec{g}$ consisting of m tokens generated using f's outputs in $\mathbf{X}_g \in \mathbb{R}^{m \times V}$ (and $\mathbf{X}_s$). We refer to $\mathbf{X}_s$ or $\mathbf{X}_g$ as *Token Distribution Sequences* (TDS). See Figure 2.

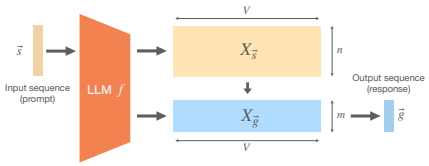

Figure 2: LLM processing pipeline. A token sequence $\vec{s}$ is processed by an LLM f and generates full TDSs $\mathbf{X}_s, \mathbf{X}_g$ corresponding to the input $\vec{s}$ and the response $\vec{g}$.

We also define $\mathbf{p}_s \in \mathbb{R}^n, \mathbf{p}_g \in \mathbb{R}^m$, which holds the probabilities associated with the actual tokens appearing in $\vec{s}, \vec{g}$ respectively. We denote these as the *Actual Token Probabilities* (ATP). Specifically, $(\mathbf{p}_s)_i := \mathbf{X}_{i,v}$ where $v \in \{1, \ldots, V\}$ is the token used in the $i+1$ place in the sequence $\vec{s}$ and similarly for $\vec{g}$. See Figure 1 for an illustration.

We call the pairs $(\mathbf{X}_s, \mathbf{p}_s)$ or $(\mathbf{X}_g, \mathbf{p}_g)$ the *LLM Output Signature* (LOS). For DCD, we analyze input sequences using $(\mathbf{X}_s, \mathbf{p}_s)$ since our interest lies in how the model processes the input text $\vec{s}$. In contrast, for HD, we use $(\mathbf{X}_g, \mathbf{p}_g)$ as we need to analyze the model's generated response. We will sometimes use $(\mathbf{X}, \mathbf{p})$ if the distinction between the tasks is irrelevant, and use N as the sequence length.

Problem Statement. LOS elements, along with their associated annotations depending on the task of interest, can be gathered into datasets $D = \{((\mathbf{X}, \mathbf{p})_i, y_i)\}_{i=1}^{\ell}$ where supervised learning problems can be instantiated. Our goal in this paper is to propose a neural architecture that can effectively utilize the complete LOS to solve tasks such as DCD, HD, or any other classification problem defined thereon.

3.2 Our Approach

Our approach consists of three main steps. Given an input $(\mathbf{X}, \mathbf{p})$: (1) The probability distributions in TDS $(\mathbf{X})$ are sorted independently and sliced to only include the top K ones at each time step, obtaining $\mathbf{X}'$; (2) A learnable Rank Encoding $\mathrm{RE}(\mathbf{X}, \mathbf{p})$ is concatenated to $\mathbf{X}'$ to capture relative

probability information; (3) The resulting representation is processed by a lightweight transformer architecture, yielding the desired output. In the remainder of this section, we provide a detailed explanation of each component.

Preprocessing the token distribution sequences. Utilizing $\mathbf{X}$ may pose significant challenges due to three key factors. *(1) Complexity:* The vocabulary tensor can be extremely large in real-world scenarios. For instance, Liang et al. (2023) (XLM-V) reported a vocabulary size of 1M tokens, which, for a small batch of documents and popular context sizes, would already entail processing a tensor of tens (or hundreds) of GBs. *(2) Transferability:* Vocabulary size and order may significantly vary between LLMs, something which can complicate transfer learning – e.g., training on one LLM and testing on another with a different vocabulary size; *(3) Limited Access:* In certain LLMs, such as those released by OpenAI, the output tensor $\mathbf{X}$ is only partially accessible, with APIs only exposing the (log-) probabilities for a small number of most likely tokens.

To tackle these challenges, we propose selecting, for each row of $\mathbf{X}$, a fixed number of elements. Specifically, we preprocess $\mathbf{X}$ by sorting each row independently and selecting the top K probabilities, as follows:

$$\mathbf{X}' = \text{row-sort}(\mathbf{X})_{:,:K}, \tag{1}$$

resulting in $\mathbf{X}' \in \mathbb{R}^{N \times K}$. This approach not only reduces computational complexity but also provides a standardized representation that is independent of the vocabulary size (for an appropriate choice of K). Later, in Section 5, we will show how our approach can achieve strong empirical performance even for small values of K. Nevertheless, it is important to note that this preprocessing step removes the alignment of words across the vocabulary dimension. Exploring methods to retain or effectively utilize this alignment remains an avenue for future work.

Learnable Rank Encoding. The tensor $\mathbf{X}'$ provides a comprehensive description of the LLM's output, but does not encode an important source of information: the probability $\mathbf{p}$ of the actual tokens appearing in the sequence, i.e, the ATP. The importance of this feature both in DCD and HD has already been demonstrated by a large body of prior work that operated only on this information Shi et al. (2023); Guerreiro et al. (2022); Kadavath et al. (2022); Varshney et al. (2023); Huang et al. (2023b). Taking inspiration from these, we do also include ATPs as inputs to our architecture. However, we further complement these probabilities with additional information which allows us to contextualize them with respect to the whole TDS, i.e., $\mathbf{X}$. Specifically, we argue that valuable information is encoded in the *rank* (position) of the ATP within the vocabulary-wide (sorted) sequence of token probabilities. This information reveals both the model's generation patterns and potential mismatches between predicted and actual tokens. The rank of the i-th token in the sequence is defined as: $r_i(\mathbf{X}, \mathbf{p}) = \sum_{v=1}^{V} \mathbb{I}(\mathbf{X}_{i,v} > p_i)$, where $\mathbb{I}(\cdot)$ is the indicator function.

We encode the rank in a way to make this feature more amenable for learning, while still maintaining enough expressivity. Specifically, we first scale the rank between $[-1, 1]$, obtaining $\mathbf{r}^{\text{scaled}}$. Then, we construct the following learnable rank encoding[2],

$$\text{RE}(\mathbf{X}, \mathbf{p}) = \mathbf{p} \otimes \left(\mathbf{r}^{\text{scaled}} \cdot \mathbf{w}_1 + \mathbf{w}_2\right), \tag{2}$$

where $\otimes$ is an outer product, and $\mathbf{w}_1, \mathbf{w}_2$ are learnable parameters in $\mathbb{R}^d$. As a result, $\text{RE}(\mathbf{X}, \mathbf{p})$ is in $\mathbb{R}^{N \times d}$. Importantly, the multiplication by $\mathbf{p}$ makes sure that the rank encoding and the TDS are in similar scales, especially when using log probabilities or logits.

Architecture. Given the preprocessed TDS $\mathbf{X}'$, and the developed rank encodings $\text{RE}(\mathbf{X}, \mathbf{p})$, our approach applies an encoder-only transformer model $\mathcal{T}$ with learnable positional encodings in the temporal dimension (Vaswani, 2017), on the token-wise linear projection of $\mathbf{X}'$, concatenated to the rank encodings $\text{RE}(\mathbf{X}', \mathbf{p})$,

$$h_\theta(\mathbf{X}, \mathbf{p}) = \mathcal{T}\left(\mathbf{X}'\mathbf{W} \,\middle\|\, \text{RE}(\mathbf{X}, \mathbf{p})\right). \tag{3}$$

Here, $\mathbf{W} \in \mathbb{R}^{K \times K'}$, $\|$ denotes concatenation along the feature dimension, and θ includes all parameters, $\mathbf{w}_1, \mathbf{w}_2, \mathbf{W}$ and the parameters of $\mathcal{T}$. We use a [CLS] token pooling mechanism at the end, followed by a linear projection for classification with standard binary cross-entropy loss. We dub the resulting model LOS-NET.

[2]For certain DC datasets, we used a lookup table for Rank encoding, where the index corresponds to r_i and the value is an embedding.

4 GENERALIZATION OF PREVIOUS APPROACHES

In this section, we demonstrate that LOS-NET generalizes several leading existing methods through specific weight configurations. This ensures that our architecture can theoretically match these methods, while in practice significantly outperforming them, as demonstrated in our experiments. As already mentioned, prior research has introduced various methods for analyzing LLMs based on their output probabilities Guerreiro et al. (2022); Kadavath et al. (2022); Varshney et al. (2023); Huang et al. (2023b), with many approaches focusing on the ATPs. We note that many of these methods assume the form of statistics calculated over the whole sequence processed by the LLM. Recent, more sophisticated approaches aggregate these probabilities only for some of the tokens in the sequence, dynamically chosen based on features computed on the set of ATPs (Shi et al., 2023; Zhang et al., 2024), as we illustrate below.

Motivating example: Min-K% Shi et al. (2023). Min-K% makes predictions on an input text $\vec{s}$ based on a score R calculated as the average of the smallest $K\%$ log-probs: $R(\vec{s}) = \frac{1}{|M|}\sum_{i\in M}\log(p_i)$, with $M = \{i \mid p_i < \text{perc}(\mathbf{p}, K)\}$ being the set of token indices whose probabilities are in the first K-th percentile of $\mathbf{p}$. We note that it is instructive to rewrite the scoring equation as:

$$R(\vec{s}) = \sum_{i=1}^{|\vec{s}|} \overbrace{\frac{\log(p_i)}{\lceil \frac{K}{100}\cdot|\vec{s}|\rceil}}^{\text{token-wise score}} \cdot \underbrace{\mathbb{I}\big(\overbrace{p_i}^{\text{confidence}} < \overbrace{\text{perc}(\mathbf{p}, K)}^{\text{adaptive threshold}}\big)}_{\text{gating}}. \tag{4}$$

This highlights a general pattern: that of computing a global score by aggregating token-wise values meeting a (dynamic) "acceptance" condition, a form of 'gating'. To unify the aforementioned baselines under a common framework, we formalize this pattern in a family of functions, defined next.

Gated Scoring Functions (GSFs). We define the family of 'Gated Scoring Functions' (GSF) as the set of functions that score LOSs by aggregating token-wise scores across the input sequence whenever their confidence values exceed a (possibly adaptive) threshold. GSFs are described in terms of the following three components: (1) A confidence function $\kappa : \mathbb{R}^{N\times k} \times \mathbb{R}^N \to \mathbb{R}^N$ that assigns confidence values to each token in the sequence; (2) A threshold function $T : \mathbb{R}^{N\times k} \times \mathbb{R}^N \to \mathbb{R}$ that determines an acceptance criterion; and (3) A weight function $g : \mathbb{R}^{N\times k} \times \mathbb{R}^N \to \mathbb{R}^N$ that assigns importance scores to tokens. Given a LOS $(\mathbf{X}, \mathbf{p})$, a GSF computes a global score $R(\mathbf{X}, \mathbf{p})$ as follows:

$$F(\mathbf{X}, \mathbf{p})_i = \begin{cases} g(\mathbf{X}', \mathbf{p})_i, & \text{if } \kappa(\mathbf{X}', \mathbf{p})_i \geq T(\mathbf{X}', \mathbf{p}), \\ 0, & \text{otherwise,} \end{cases} \tag{5}$$

$$R(\mathbf{X}, \mathbf{p}) = \sum_{i=1}^{N} F(\mathbf{X}, \mathbf{p})_i, \tag{6}$$

Where $\mathbf{X}'$ is the sorted version of $\mathbf{X}$, as per Equation (1). The family of GSF is flexible enough to capture previously proposed gray-box methods, as we show in the following:

Proposition 4.1 (GSFs capture known baselines)**.** *Let $\mathcal{B}$ be the set of scoring functions implemented by the Min/Max/Mean aggregated probability methods (Guerreiro et al., 2022; Kadavath et al., 2022; Varshney et al., 2023; Huang et al., 2023b) for HD, as well as the MinK% (Shi et al., 2023) and MinK%++ (Zhang et al., 2024) methods for DCD. For any scoring function $f \in \mathcal{B}$, there exists a choice of functions κ, T, g such that the GSF R in Equation* (6)*, implements f.*

It is easy to see, e.g., how MinK% is implemented as a GSF. For a sequence length of N, it suffices to choose:

$$T(\mathbf{X}', \mathbf{p}) = -\text{perc}(\mathbf{p}, K) = -\big(\text{sort}(\mathbf{p})_{\lceil \frac{K}{100}\cdot N\rceil}\big),$$

$$\kappa(\mathbf{X}', \mathbf{p}) = -\mathbf{p}, \quad g(\mathbf{X}', \mathbf{p}) = \frac{\log \mathbf{p}}{\lceil \frac{K}{100}\cdot N\rceil}.$$

We refer readers to Appendix B for a proof of Proposition 4.1 and more details on how other baselines are implemented.

LOS-NET can approximate GSFs and implement known baselines. As the following results show, our LOS-NET architecture is theoretically justified from an expressiveness standpoint. We start by showing that it can, in fact, approximate virtually all GSFs of interest.

Proposition 4.2 (Our model can approximate Equation (6))**.** *Assume a maximal possible vocabulary size V_{max} and a maximal context size N_{max}. Let $\mathcal{X} \times \mathcal{M} \subseteq \mathbb{R}^{N_{max} \times V_{max}} \times \mathbb{R}^{N_{max}}$ represent a compact subset in the LOS. For any measurable $\kappa : \mathcal{X} \times \mathcal{M} \to \mathbb{R}^{N_{max}}$, measurable $T : \mathcal{X} \times \mathcal{M} \to \mathbb{R}$, measurable and integrable weight function $g : \mathcal{X} \times \mathcal{M} \to \mathbb{R}^{N_{max}}$, and for any $\epsilon > 0$, there exists a set of parameters θ such that our model $h_\theta : \mathcal{X} \times \mathcal{M} \to \mathbb{R}$ satisfies $\|h_\theta - R\|_{L_1} < \epsilon$ where $\|\cdot\|_{L_1}$ denotes the L_1 norm.*

The complete proof is given in Appendix B. To prove this result, we build on existing universality results for approximating continuous functions with Transformers (Yun et al., 2019), by showing that our (generally non-continuous) target functions can be approximated by continuous functions. The implications of this proposition are interesting: as long as the LOS space of interest lies within a compact domain – an assumption inherently satisfied when using, e.g., probabilities[3] – our model can approximate the general GSF given in Equation (6) of LOSs of any LLM under mild conditions on κ, T, and g, potentially generalizing across LLMs as our approximation result considers a predefined LOS domain. In Section 5 we show that our trained models can indeed be applied successfully out-of-the-box on LOSs from different LLMs. We note that Proposition 4.2 cannot be generally extended to L_∞ due to the discontinuity of GSFs. The practical relevance of Proposition 4.2, is underscored by the following corollary:

Corollary 4.3 (Approximation of Baselines by LOS-NET)**.** *Our architecture, as defined in Equation* (3)*, can arbitrarily well approximate, in the L_1 sense, any of the baseline methods in $\mathcal{B}$ when operating on context and token-vocabulary of, resp., maximal sizes N_{max} and V_{max}.*

The above states that well-established, successful baselines from the literature (see class $\mathcal{B}$ in Proposition 4.1) can be approximated by LOS-NET. The proof for Corollary 4.3 follows from Propositions 4.1 and 4.2, see Appendix B for the complete proof.

5 EXPERIMENTS

We assess various aspects of learning with LOS via the following questions: **(Q1)** Is learning on LOS an effective approach for addressing key tasks such as DCD and HD? Does it outperform baselines? (Sections 5.1 and 5.2); **(Q2)** Does our model exhibit transfer capabilities across LLMs and across datasets, suggesting the emergence of universal patterns in LLM behavior from the LOS perspective? (Section 5.3, and Appendix C.7); **(Q3)** How important is $\mathbf{X}$ in the pair $(\mathbf{X}, \mathbf{p})$, as it is often overlooked? And how impactful is the choice of the slicing parameter K in Equation (1) (Appendix C.6). In the following, we present our main results, and refer to Appendix C for additional experiments, and experimental details.

General setup. Our experiments focus on the two tasks of DCD and HD, with hyperparameter K fixed at 1000 unless stated otherwise (see Equation (1)). In Appendix C.6, we show that our model is robust to variations in K. To align with prior work, we use datasets and LLMs from Shi et al. (2023); Zhang et al. (2024) for DCD and Orgad et al. (2024) for HD, totaling six datasets – three for DCD and three for HD – and seven LLMs, five for DCD and two for HD. Further details are in subsequent sections. We use the area under the ROC curve (AUC) to evaluate HD and DCD, a standard metric in this domain Orgad et al. (2024); Shi et al. (2023); Zhang et al. (2024), which measures the balance between sensitivity and specificity. We conduct each experiment across three different random seeds (when applicable) and report the mean along with the standard deviation of the results.

Newly introduced learning-based baselines. In addition to task-specific baselines, detailed in the following, we also introduce two novel learning-based baselines to appreciate the contribution of TDS. In these baselines, which we call ATP+Rank-MLP and ATP+Rank-Transformer (dubbed ATP+R-MLP, ATP+R-TRANSF., respectively), we ablate information about the TDS: they only process Rank Encodings (Equation (2)), thus accessing the ATP and rank information only. Formal definitions are in Appendix C.4.

5.1 HALLUCINATION DETECTION

We follow the setup of Orgad et al. (2024). The objective is to predict whether an LLM-generated response to a given input prompt is correct or not. We frame the task within a gray-box setting,

[3]For logits or log probabilities, clamping ensures the compactness assumption.

Table 1: Comparison of AUC over Mis-7b and L3-8b on HD, across the discussed baseline methods. The best-performing method is in **bold**, and the second best is underlined.

Method	HotpotQA	IMDB	Movies	HotpotQA	IMDB	Movies
	Mistral-7b-instruct			Llama3-8b-instruct		
Logits-mean	61.00 ± 0.20	57.00 ± 0.60	63.00 ± 0.50	65.00 ± 0.20	59.00 ± 1.70	75.00 ± 0.50
Logits-min	61.00 ± 0.30	52.00 ± 0.70	66.00 ± 0.80	67.00 ± 0.80	55.00 ± 1.60	71.00 ± 0.50
Logits-max	53.00 ± 0.80	47.00 ± 0.40	54.00 ± 0.40	59.00 ± 0.50	51.00 ± 0.90	67.00 ± 0.30
Probas-mean	63.00 ± 0.30	54.00 ± 0.80	61.00 ± 0.20	61.00 ± 0.20	73.00 ± 1.50	73.00 ± 0.60
Probas-min	58.00 ± 0.30	51.00 ± 1.00	60.00 ± 0.80	60.00 ± 0.40	57.00 ± 1.60	65.00 ± 0.40
Probas-max	50.00 ± 0.50	48.00 ± 0.40	51.00 ± 0.50	56.00 ± 0.50	49.00 ± 0.80	64.00 ± 0.60
P(True)	54.00 ± 0.60	62.00 ± 0.90	62.00 ± 0.50	55.00 ± 0.50	60.00 ± 0.60	66.00 ± 0.40
ATP+R-MLP	61.36 ± 0.33	88.95 ± 0.40	60.63 ± 0.16	60.09 ± 0.24	85.28 ± 0.49	67.19 ± 0.25
ATP+R-TRANSF.	63.78 ± 0.98	92.30 ± 1.66	62.41 ± 0.22	61.39 ± 1.24	82.56 ± 0.63	64.95 ± 0.68
LOS-NET	**73.24** ± 0.28	**96.11** ± 0.03	**68.59** ± 1.08	**72.97** ± 0.41	**89.44** ± 0.32	**77.04** ± 0.77

Method / LLM	P-6.9b	P-12b	L-13b	L-30b
Loss	67.40	76.27	76.23	89.18
MinK	68.78	77.32	75.36	89.61
MinK++	66.73	71.76	72.87	80.60
Zlib	50.01	60.84	61.94	80.83
Lowercase	74.97	81.64	67.80	82.18
Ref	89.52	**91.93**	84.58	94.93
ATP+R-MLP	56.31 ± 1.48	57.18 ± 1.06	66.60 ± 1.05	83.89 ± 0.41
ATP+R-TRANSF.	79.59 ± 0.61	74.77 ± 0.57	74.65 ± 0.79	87.62 ± 0.68
LOS-NET	**90.71** ± 0.90	89.43 ± 0.59	**91.02** ± 0.15	**95.60** ± 0.41

Table 2: Test AUC on BookMIA. 'P': Pythia, 'L': LLaMa-1. Best result is in **Bold**, second best is underlined. Reference-based approaches are shaded in pink.

i.e., we assume no access to the LLM's internals. We also assume no access to external resources, e.g., other LLMs, auxiliary, repeated prompting, or any additional contextual information, such as pointers to specific answer tokens. This makes methods like Orgad et al. (2024); Kuhn et al. (2023) not directly comparable. Note, however, that these additional sources of information could easily be incorporated into LOS-NET, e.g., using a one-hot vector to flag specific answer tokens or by extending the LOS to account for additional prompting. We defer investigating these more relaxed settings to future research.

Datasets and LLMs. Following Orgad et al. (2024), we use three datasets spanning various domains and tasks: HotpotQA without context Yang et al. (2018), IMDB sentiment analysis Maas et al. (2011), and movie roles Orgad et al. (2024). Further details, regarding annotations' collection process, the splits and dataset sizes are in Appendix C.5.1. As the target LLMs, coherently with Orgad et al. (2024), we use Mistral-7b-instruct-v0.2 Jiang et al. (2023) (Mis-7b), and LLaMa3-8b-instruct Touvron et al. (2023) (L-3-8b).

HD Baselines. As baselines for hallucination detection, we consider the following, (1) Aggregated probabilities/logits: Previous studies Guerreiro et al. (2022); Kadavath et al. (2022); Varshney et al. (2023); Huang et al. (2023b) simply aggregate output token probabilities or logits to score LLM confidence for error detection. Such simple aggregations include mean/max/min over the ATP. We refer to them as Logit/Probas-mean/min/max; (2) P(True): Kadavath et al. (2022) found that LLMs show reasonable calibration in assessing their own output correctness.

Results. Table 1 presents a comprehensive summary of our main results, which clearly demonstrate that LOS-NET outperforms all baselines across all six dataset/LLM combinations, often by a significant margin. For instance, on the IMDB dataset, LOS-NET achieves an AUC improvement of around 34 units over the best baseline for Mis-7b and 16 over the best baseline for L-3-8b-instruct. Our results further indicate that ATP learning-based baselines consistently underperform compared to LOS-NET, underscoring the critical role of the TDS, $\mathbf{X}$, in achieving superior results. However, our ATP-based learnable baselines outperform Probas/Logits-based baselines in 3 out of 6 cases, suggesting that a learning approach relying exclusively on ATP can still be a viable solution in certain scenarios.

5.2 DATA CONTAMINATION DETECTION

The goal in DCD is to determine if an LLM was trained on specific data. The raw dataset $D = \{q_i, y_i\}_{i=1}^{\ell}$ contains ℓ text samples, where q_i represents the text and y_i indicates whether it was part of the training data. DCD is often framed as a Membership Inference Attack (MIA) (Shokri et al., 2017; Mattern et al., 2023; Shi et al., 2023).

Table 3: Comparison of AUC over four different LLMs, on DCD, over the discussed baselines methods. The best-performing method is in **bold**, and the second best is underlined. Reference-based approaches are shaded in pink.

Dataset →	WikiMIA - 32				WikiMIA - 64			
LLM →	P-6.9b	L-13b	L-30b	M-1.4b	P-6.9b	L-13b	L-30b	M-1.4b
Loss	63.82 ±2.22	67.45 ±1.57	69.37 ±2.66	60.89 ±1.35	60.59 ±3.50	63.68 ±5.57	66.18 ±4.64	58.46 ±3.69
MinK	66.39 ±2.56	68.08 ±1.45	70.02 ±2.92	63.27 ±1.85	65.07 ±1.80	66.24 ±5.01	68.45 ±4.11	62.46 ±2.75
MinK++	70.60 ±3.58	84.93 ±1.76	84.46 ±1.43	67.06 ±2.78	71.82 ±3.73	85.66 ±2.25	85.02 ±2.79	67.24 ±4.06
Zlib	64.35 ±3.46	67.70 ±2.25	69.81 ±3.17	62.07 ±3.35	62.59 ±3.38	65.40 ±5.35	67.61 ±4.21	60.59 ±3.73
Lowercase	62.09 ±4.22	64.03 ±6.97	64.31 ±5.18	60.59 ±3.24	58.34 ±4.21	62.63 ±5.05	61.54 ±7.81	57.03 ±2.83
Ref	63.45 ±6.03	57.77 ±5.94	63.55 ±6.69	62.05 ±5.43	62.35 ±4.84	63.07 ±5.09	68.94 ±5.83	60.29 ±4.62
LOS-NET	**76.98** ±3.36	**93.46** ±1.31	**93.76** ±1.56	**71.04** ±9.07	**76.00** ±5.48	**87.86** ±3.73	**93.04** ±2.51	**79.39** ±2.61

Datasets and LLMs. We use three datasets to assess DCD, specifically: WikiMIA-32 and WikiMIA-64 Shi et al. (2023), as well as BookMIA Shi et al. (2023). The WikiMIA-32 and -64 datasets contain excerpts from Wikipedia articles, consisting of, resp., 32 and 64 words. The distinction between contaminated and uncontaminated data is determined by timestamps. As in (Shi et al., 2023; Zhang et al., 2024), we attack Mamba-1.4b Gu & Dao (2023) (M-1.4b), LLaMa-13b/30b Touvron et al. (2023) (L-13b/30b), Pythia-6.9b Biderman et al. (2023) (P-6.9b). BookMIA is a dataset of book excerpts. Positive members correspond to books known to be well memorized by certain OpenAI models (Chang et al., 2023), or otherwise known to (partly) be in pretraining corpus of other open-source LLMs (Antebi et al., 2025). Non-members include excerpts from books released after 2023, necessarily absent from the pretraining corpus of the these last ones. Interestingly, this dataset allows us to test LOS-NET's DCD capability in a realistic scenario akin to copyright-infringement detection. We thus propose a new split that ensures all excerpts from the same book always appear either in the training or test split (and never in both). Details are enclosed in Appendix C.5.2. We attack LLMs considered in (Antebi et al., 2025): LLaMa-13b/30b Touvron et al. (2023) (L-13b/30b), Pythia-6.9b/12bBiderman et al. (2023) (P-6.9b/12b).

DCD Baselines. We evaluate six recent methods as our baselines. The Loss approach Yeom et al. (2018) directly uses the loss value as the detection score. The Reference (Ref) method Carlini et al. (2021) calibrates the target LLM's perplexity leveraging a similar reference model known or supposed not to have memorized text of interest[4]. Both Zlib and Lowercase Carlini et al. (2021) are also reference-based methods: they utilize zlib compression entropy and lowercased text perplexity as reference for normalization. Lastly, Min-K% Shi et al. (2023) and Min-K%++ Zhang et al. (2024) are reference-free methods, which examine token probabilities and average a subset of the minimum token scores, or a function thereof, over the input. Min-K%++ is currently the best-performing method on the WikiMIA dataset. For these baselines, we select their hyperparameters by maximizing performance on the validation set(s).

Results on BookMIA. Results are reported in Table 2. On this benchmark, our method attains exceptional results, largely surpassing other reference-free approaches. Among these last ones, ours is the only method that can match or outperform the reference-LLM-based baselines. Additionally, our experimental results suggest that instrumental to achieve such strong reference-free performance is to access, even partially, the TDS, as our ATP-based learnable baselines, which only process features for the actual sequence tokens, incur significant performance degradations.

Results on WikiMIA. Since WikiMIA does not provide an official training split and our method requires labeled data, we perform 5-fold cross-validation with training, validation, and testing splits[5] and rerun all baselines under the same protocol for a fair comparison. Results are reported as the mean and standard deviation across folds. For these datasets only, setting the hyperparameter $K = 1000$ (recall Equation (1)) led to suboptimal performance in preliminary experiments, thus, we set $K =$ "Full-Vocabulary". As shown in Table 3, LOS-NET consistently surpasses all baseline methods across all eight combinations of LLMs and datasets. Notably, for L-30b, our model achieves an AUC score that is more than 8 points higher than the best-performing baseline, MinK%++ for both datasets, demonstrating a substantial improvement. Similarly, for P-6.9b, our model maintains a steady advantage of approximately 5 AUC for both datasets, further underscor-

[4]For example for Pythia-12b, a valid reference LLM would be the smaller Pythia-70M.

[5]We use $\{\frac{3}{5}, \frac{1}{5}, \frac{1}{5}\}$ as the ratios for training, validation, and testing, respectively.

ing its robustness. Overall, the second-best method is MinK%++, followed by MinK%, consistently with the findings of Zhang et al. (2024).

5.3 Generalization to other LLMs and datasets

Here, we further study the possibility to apply our models to settings different from those they were originally trained on. We focus on two variables: datasets and LLMs. Generalization across different datasets was originally studied in (Orgad et al., 2024) within the scope of HD, and in the context of white-box setups; their relevance lies in the fact that non-trivial dataset generalization would potentially suggest a 'universal truthfulness' representation encoded in the internal states and/or outputs of an LLM (Orgad et al., 2024; Marks & Tegmark, 2023; Slobodkin et al., 2023). On the other hand, inspecting transfer across LLMs is, to the best of our knowledge, still unexplored. This study would be important for learning based approaches for applications such as copyright-infringement detection, where ground-truth labels may be scarce.

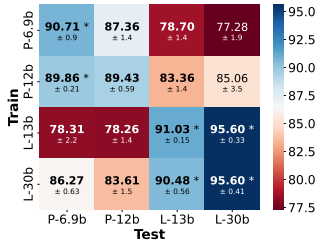

Figure 3: BookMIA zero-shot generalization. **Bold**: highest score among reference-free baselines; superscript *: LOS-Net also surpasses reference-based methods.

Zero Shot Cross-LLM Generalization Capabilities in DCD. We assess our model's ability to detect DC in target LLMs that were unseen during training. Using the BookMIA benchmark and the setup described in Section 5.2, we evaluate our model directly across different LLMs *without any fine-tuning*. This setup is relevant in cases where contamination information is not yet available for newly released LLMs. The results are presented in the heatmap shown in Figure 3. We observe strong transferability: in 10/12 cases, our model achieves the best performance among reference-free approaches, highlighted in bold in Figure 3. Interestingly, in 3/12 cases, LOS-Net (which is reference-free) even surpasses reference-based baselines, as indicated via a superscript of *. We also observe particularly strong transfer across differently sized LLM architectures within the same family and highlight the surprising positive transfer from the largest LLaMa to Pythia models.

Discussion. We observe several key findings. First, LOS-Net exhibits solid transferability in both scenarios. The finetuned models consistently outperform their counterparts trained from scratch: 6/6 cases in the cross-LLM setup (Figure 6), 11/12 cases in the cross-dataset one (Figure 7), as indicated via * on the off-diagonal entries; the full results for training from scratch are presented in Figures 8 and 9 in the Appendix. This highlights the effectiveness of LOS as a data type in capturing universal patterns in LLM behavior. Second, from a practical perspective, we find that LOS-Net outperforms the best baseline in 4/6 cases for the cross-LLM scenario (Figure 6) and in 9/12 cases for the cross-dataset scenario (Figure 7), as indicated in bold on the off-diagonal entries. Focusing on the IMDB dataset, when training on L-3-8b and testing on Mis-7b (Figure 6), our model achieves a substantial gain of around 29 AUC units over the best baseline. This result underscores the possibility of transferring across LLMs. A similar trend is observed in the cross-dataset setup (Figure 7): on Mis-7b, when training on HotpotQA or Movies and testing on IMDB, our model achieves a notable improvement of around 31, 27 AUC units, respectively, compared to the best baseline.

6 Conclusion

We proposed LOS-Net, an efficient method for Data Contamination and Hallucinations Detection (DCD, HD) in Large Language Models (LLMs) by leveraging their output signatures (LOS). LOS-Net processes the Token Distribution Sequence (TDS) and Actual Token Probabilities (ATP) using a lightweight transformer with learnable rank encoding, capturing richer contextual information. We theoretically showed that LOS-Net unifies and extends existing gray-box methods under a general framework. Experiments across datasets and LLMs show LOS-Net outperforms state-of-the-art gray-box baselines in HD and DCD. We demonstrated strong generalization capabilities of LOS-Net, both across datasets and across LLMs, where the latter suggests that LOS-Net can effectively capture universal patterns in LLM behavior. Several avenues for future research remain open. Our framework could extend beyond DCD and HD to other tasks, such as detecting LLM-generated content. Exploring more complex architectures than LOS-Net is also interesting, for example, we note that sorting the TDS tensor removes word alignment across the vocabulary, which may be limiting in some cases.

ACKNOWLEDGMENTS

The authors are grateful to Beatrice Bevilacqua for insightful discussions. G.B. is supported by the Jacobs Qualcomm PhD Fellowship. F.F. conducted this work supported by an Aly Kaufman and an Andrew and Erna Finci Viterbi Post-Doctoral Fellowship. Y.G. is supported by the UKRI Engineering and Physical Sciences Research Council (EPSRC) CDT in Autonomous and Intelligent Machines and Systems (grant reference EP/S024050/1). H.M. is a Robert J. Shillman Fellow and is supported by the Israel Science Foundation through a personal grant (ISF 264/23) and an equipment grant (ISF 532/23). D.L. is funded by an NSF Graduate Fellowship. Research was also supported by the Israeli Ministry of Science, Israel-Singapore binational grant 207606.

REFERENCES

Sagiv Antebi, Edan Habler, Asaf Shabtai, and Yuval Elovici. Tag&tab: Pretraining data detection in large language models using keyword-based membership inference attack. *arXiv preprint arXiv:2501.08454*, 2025.

Yuval Atzmon and Gal Chechik. Adaptive confidence smoothing for generalized zero-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11671–11680, 2019.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

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

Lukas Biewald. Experiment tracking with weights and biases, 2020. URL `https://www.wandb.com/`. Software available from wandb.com.

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. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

Meng Cao, Yue Dong, Jingyi He, and Jackie Chi Kit Cheung. Learning with rejection for abstractive text summarization. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 9768–9780, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.663. URL `https://aclanthology.org/2022.emnlp-main.663/`.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX security symposium (USENIX security 19)*, pp. 267–284, 2019.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914. IEEE, 2022.

Kent Chang, Mackenzie Cramer, Sandeep Soni, and David Bamman. Speak, memory: An archaeology of books known to ChatGPT/GPT-4. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7312–7327, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.453. URL https://aclanthology.org/2023.emnlp-main.453/.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Nuno M Guerreiro, Elena Voita, and André FT Martins. Looking for a needle in a haystack: A comprehensive study of hallucinations in neural machine translation. *arXiv preprint arXiv:2208.05309*, 2022.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. *arXiv preprint arXiv:1909.03368*, 2019.

John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4129–4138, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1419. URL https://aclanthology.org/N19-1419/.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 2023a.

Yuheng Huang, Jiayang Song, Zhijie Wang, Shengming Zhao, Huaming Chen, Felix Juefei-Xu, and Lei Ma. Look before you leap: An exploratory study of uncertainty measurement for large language models. *arXiv preprint arXiv:2307.10236*, 2023b.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251): 1–43, 2023.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation, 2023. URL https://arxiv.org/abs/2302.09664.

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

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36, 2024a.

Yucheng Li, Yunhao Guo, Frank Guerin, and Chenghua Lin. An open-source data contamination report for large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 528–541, Miami, Florida, USA, November 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.30. URL https://aclanthology.org/2024.findings-emnlp.30/.

Davis Liang, Hila Gonen, Yuning Mao, Rui Hou, Naman Goyal, Marjan Ghazvininejad, Luke Zettlemoyer, and Madian Khabsa. Xlm-v: Overcoming the vocabulary bottleneck in multilingual masked language models. *arXiv preprint arXiv:2301.10472*, 2023.

Tianyu Liu, Yizhe Zhang, Chris Brockett, Yi Mao, Zhifang Sui, Weizhu Chen, and Bill Dolan. A token-level reference-free hallucination detection benchmark for free-form text generation. *arXiv preprint arXiv:2104.08704*, 2021.

I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150, 2011.

Samuel Marks and Max Tegmark. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *arXiv preprint arXiv:2310.06824*, 2023.

Justus Mattern, Fatemehsadat Mireshghallah, Zhijing Jin, Bernhard Schölkopf, Mrinmaya Sachan, and Taylor Berg-Kirkpatrick. Membership inference attacks against language models via neighbourhood comparison. *arXiv preprint arXiv:2305.18462*, 2023.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1906–1919, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.173. URL https://aclanthology.org/2020.acl-main.173/.

Edoardo Mosca, Shreyash Agarwal, Javier Rando Ramírez, and Georg Groh. "that is a suspicious reaction!": Interpreting logits variation to detect NLP adversarial attacks. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7806–7816, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.538. URL https://aclanthology.org/2022.acl-long.538/.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. Llms know more than they show: On the intrinsic representation of llm hallucinations. *arXiv preprint arXiv:2410.02707*, 2024.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Yifu Qiu, Yftah Ziser, Anna Korhonen, Edoardo Ponti, and Shay Cohen. Detecting and mitigating hallucinations in multilingual summarisation. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 8914–8932, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.551. URL https://aclanthology.org/2023.emnlp-main.551/.

Yifu Qiu, Zheng Zhao, Yftah Ziser, Anna Korhonen, Edoardo M Ponti, and Shay B Cohen. Spectral editing of activations for large language model alignment. *arXiv preprint arXiv:2405.09719*, 2024.

Miriam Rateike, Celia Cintas, John Wamburu, Tanya Akumu, and Skyler Speakman. Weakly supervised detection of hallucinations in llm activations. *arXiv preprint arXiv:2312.02798*, 2023.

Vipula Rawte, Swagata Chakraborty, Agnibh Pathak, Anubhav Sarkar, SM Tonmoy, Aman Chadha, Amit P Sheth, and Amitava Das. The troubling emergence of hallucination in large language models–an extensive definition, quantification, and prescriptive remediations. *arXiv preprint arXiv:2310.04988*, 2023.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2023.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pp. 3–18. IEEE, 2017.

Aviv Slobodkin, Omer Goldman, Avi Caciularu, Ido Dagan, and Shauli Ravfogel. The curious case of hallucinatory (un) answerability: Finding truths in the hidden states of over-confident large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3607–3625, 2023.

Ben Snyder, Marius Moisescu, and Muhammad Bilal Zafar. On early detection of hallucinations in factual question answering. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2721–2732, 2024.

SM Tonmoy, SM Zaman, Vinija Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv preprint arXiv:2401.01313*, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Simon Valentin, Jinmiao Fu, Gianluca Detommaso, Shaoyuan Xu, Giovanni Zappella, and Bryan Wang. Cost-effective hallucination detection for llms. *arXiv preprint arXiv:2407.21424*, 2024.

Neeraj Varshney, Wenlin Yao, Hongming Zhang, Jianshu Chen, and Dong Yu. A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation. *arXiv preprint arXiv:2307.03987*, 2023.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Vivek Verma, Eve Fleisig, Nicholas Tomlin, and Dan Klein. Ghostbuster: Detecting text ghostwritten by large language models. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1702–1717, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.95. URL https://aclanthology.org/2024.naacl-long.95/.

Kangxi Wu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-Seng Chua. Llmdet: A third party large language models generated text detection tool. *arXiv preprint arXiv:2305.15004*, 2023.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pp. 268–282. IEEE, 2018.

Fan Yin, Jayanth Srinivasa, and Kai-Wei Chang. Characterizing truthfulness in large language model generations with local intrinsic dimension. *arXiv preprint arXiv:2402.18048*, 2024.

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? *arXiv preprint arXiv:1912.10077*, 2019.

Jingyang Zhang, Jingwei Sun, Eric Yeats, Yang Ouyang, Martin Kuo, Jianyi Zhang, Hao Frank Yang, and Hai Li. Min-k%++: Improved baseline for detecting pre-training data from large language models. *arXiv preprint arXiv:2404.02936*, 2024.

Zheng Zhao, Emilio Monti, Jens Lehmann, and Haytham Assem. Enhancing contextual understanding in large language models through contrastive decoding. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4225–4237, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.237. URL https://aclanthology.org/2024.naacl-long.237/.

A Appendix

B Proofs

Proposition B.1 (Our model can approximate Equation (6))**.** *Assume a maximal possible vocabulary size V_{max} and a maximal context size N_{max}. Let $\mathcal{X} \times \mathcal{M} \subseteq \mathbb{R}^{N_{max} \times V_{max}} \times \mathbb{R}^{N_{max}}$ represent a compact subset in the LOS. For any measurable $\kappa : \mathcal{X} \times \mathcal{M} \to \mathbb{R}^{N_{max}}$, measurable $T : \mathcal{X} \times \mathcal{M} \to \mathbb{R}$, measurable and integrable weight function $g : \mathcal{X} \times \mathcal{M} \to \mathbb{R}^{N_{max}}$, and for any $\epsilon > 0$, there exists a set of parameters θ such that our model $h_\theta : \mathcal{X} \times \mathcal{M} \to \mathbb{R}$ satisfies $\|h_\theta - R\|_{L_1} < \epsilon$ where $\|\cdot\|_{L_1}$ denotes the L_1 norm.*

Proof. We define $\mathcal{D} \coloneqq \mathcal{X} \times \mathcal{M}$. Recall that the target function we want to approximate is the gated scoring function R as defined in Equation (6), which can be written as follows:

$$R(x) = \sum_{i=1}^{N_{\max}} \mathbb{I}(\kappa(x)_i \geq T(x)) \cdot g(x)_i, \tag{7}$$

for $x \in \mathcal{D}$.

Define $f^{(1)} : \mathcal{D} \to \mathbb{R}^{N_{\max}}$ to be the components of the sum in Equation (7):

$$f^{(1)}(x)_i = \mathbb{I}(\kappa(x)_i \geq T(x)) \cdot g(x)_i. \tag{8}$$

It follows that $R(x) = \sum_{i=1}^{N_{\max}} f^{(1)}(x)_i$.

Step 1: We begin by selecting $K = V_{\max}$ as a hyperparameter[6] and initializing the parameters $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{W}$ as follows:

$$\mathbf{p}_1 = 0, \tag{9}$$

$$\mathbf{p}_2 = 1, \tag{10}$$

$$\mathbf{W} = I_{K \times K}. \tag{11}$$

As a result, the input to the transformer encoder in our architecture (see Equation (3)) becomes $\mathbf{X}'||\mathbf{p} \in \mathbb{R}^{N_{\max} \times (V_{\max}+1)}$.

This simplifies our architecture in Equation (3) to:

$$h_\theta(\mathbf{X}, \mathbf{p}) = \mathcal{T}(\mathbf{X}'||\mathbf{p}). \tag{12}$$

Step 2: $\mathbf{f^{(1)}} \in \mathbf{L^1}(\mathcal{D})$. Define the $L^1(\mathcal{D})$ norm for a field $\mathcal{F} : \mathcal{D} \to \mathbb{R}^{n_2}$ as:

$$\|\mathcal{F}\|_{L^1} = \int_{x \in \mathcal{D}} \|\mathcal{F}(x)\|_1 \, dx = \int_{x \in \mathcal{D}} \sum_{i=1}^{n_2} |\mathcal{F}(x)_i| \, dx = \sum_{i=1}^{n_2} \int_{x \in \mathcal{D}} |\mathcal{F}(x)_i| \, dx = \sum_{i=1}^{n_2} \|\mathcal{F}(x)_i\|_{L^1}, \tag{13}$$

[6]For LLMs with a vocabulary size smaller than $V_{\max}$, appropriate padding can be applied.

where $\|v\|_1 = \sum_{i=1}^{n_2} |v_i|$ is the l_1 norm of the vector v.

Next, observe that $f^{(1)} \in L^1(\mathcal{D})$. To see this, first note that $f^{(1)}$ is measurable. The indicator function is measurable because the indicator set is the preimage of the measurable function $\kappa(x) - T(x)$ on the closed set $[0, \infty)$. Thus, $f^{(1)}$, being a product of measurable functions, is measurable. Next, we show that the L^1 norm is finite. This is true because $f^{(1)}$ is a product of the integrable function g and the bounded function $\mathbf{1}$ on the compact domain $\mathcal{D}$.

Step 3: Approximating $\mathbf{f^{(1)}}$ by a continuous field $\tilde{\mathbf{f}}^{(1)}$. We need to approximate the field $f^{(1)} : \mathcal{D} \to \mathbb{R}^{N_{\text{max}}}$ by a continuous field, so that we can apply existing results on approximating continuous functions with Transformers. We state the following Lemma, saying the continuous fields are dense in $L_1(\mathcal{D})$.

Lemma B.2. *For any $g \in L^1(\mathcal{D})$ and any $\epsilon > 0$, there exists a continuous $\tilde{g} \in L^1(\mathcal{D})$ such that $\|g - \tilde{g}\|_{L^1} < \epsilon$.*

Proof. Consider the coordinate functions $g_i : \mathcal{D} \to \mathbb{R}$. Since continuous functions are dense in L^1 for scalar valued functions, we can choose continuous $\tilde{g}_i$ such that $\|g - \tilde{g}\|_{L^1} < \epsilon/N$. Thus, letting $\tilde{g}(x) = [g_1(x), \ldots, g_N(x)] \in \mathbb{R}^N$, it holds that $\|g - \tilde{g}\| = \sum_{i=1}^{N} \|g_i - \tilde{g}_i\| < \epsilon$. □

Thus, we can choose a function $\tilde{f}^{(1)}$ such that,

$$\left\| f^{(1)} - \tilde{f}^{(1)} \right\| < \frac{\epsilon}{2N_{\text{max}}}. \tag{14}$$

Step 4: Approximating the continuous field $\tilde{\mathbf{f}}^{(1)}$ by a transformer model $\mathbf{h}_\theta^{(1)}$. We start by restating the following from Yun et al. (2019) in our context,

Theorem B.3. *Let $1 \leq p < \infty$ and $\epsilon > 0$, then for any given $f \in \mathcal{F}_{CD}$, where $\mathcal{F}_{CD}$ is the set of all continuous functions that map a compact domain in $\mathbb{R}^{n \times d}$ to $\mathbb{R}^{n \times d}$, there exists a Transformer network (with positional encodings) $g : \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times d}$ such that we have $\|f - g\|_{L^p} \leq \epsilon$.*

To apply this theorem in our context, we observe that in our case $d \coloneqq V_{\text{max}} + 1$ and $n \coloneqq N_{\text{max}}$ for the input space, and the domain $\mathcal{D} \subseteq \mathbb{R}^{N_{\text{max}} \times (V_{\text{max}}+1)}$ is compact. Thus $\tilde{f}^{(1)} \in \mathcal{F}_{\text{CD}}$ (note that the output space dimension in our case is $\mathbb{R}^{N_{\text{max}} \times 1}$ instead of $\mathbb{R}^{N_{\text{max}} \times d}$, but this can be handled using zero-padding). Using $p = 1$, it holds that there exists a transformer $h_\theta^{(1)}$ s.t., $\left\| h_\theta^{(1)} - \tilde{f}^{(1)} \right\| < \frac{\epsilon}{2N_{\text{max}}}$.

Step 5: Pooling. Our model concludes with a [CLS] token pooling mechanism, which is equivalent in expressiveness to the standard sum pooling method. Thus, assuming that the final layer of our model is given by $h_\theta^{(1)}(x)$, our model can be written as follows,

$$h_\theta(x) = \sum_{i=1}^{N_{\text{max}}} \left(h_\theta^{(1)}(x)_i \right). \tag{15}$$

Step 6: Approximating the objective function. Intuitively, $h_\theta(x)$ approximates $R(x)$ because $h_\theta^{(1)}(x)_i$ approximates $f^{(1)}(x)_i$.

We demonstrate this as follows.

$$\|h_\theta - R\|_{L_1} = \left\| \sum_{i=1}^{N_{\max}} \left(h_{\theta;i}^{(1)} \right) - \sum_{i=1}^{N_{\max}} f_i^{(1)} \right\|_{L_1} \tag{16}$$

$$\leq \sum_{i=1}^{N_{\max}} \left\| h_{\theta;i}^{(1)} - f_i^{(1)} \right\| \tag{17}$$

$$= \sum_{i=1}^{N_{\max}} \left\| h_{\theta;i}^{(1)} + (\tilde{f}_i^{(1)} - \tilde{f}_i^{(1)}) - f_i^{(1)} \right\| \tag{18}$$

$$\leq \sum_{i=1}^{N_{\max}} \left\| h_{\theta;i}^{(1)} - \tilde{f}_i^{(1)} \right\| + \sum_{i=1}^{N_{\max}} \left\| \tilde{f}_i^{(1)} - f_i^{(1)} \right\| \tag{19}$$

We applied the triangle inequality to obtain the two inequalities. Next, note that for a field $\mathcal{F} : \mathbb{R}^{n_1} \to \mathbb{R}^{n_2}$, the L^1 norm of any coordinate function is less than the L^1 norm of $\mathcal{F}$: $\|\mathcal{F}_j\|_{L^1} \leq \|\mathcal{F}\|_{L^1}$ for any $j \in \{1, \ldots, n_2\}$. This can be seen directly from the definition of the L^1 norm of $\mathcal{F}$. Combining this with our choices of $\tilde{f}$ and h_θ shows that:

$$\sum_{i=1}^{N} \left\| h_{\theta;i}^{(1)} - \tilde{f}_i^{(1)} \right\| + \sum_{i=1}^{N_{\max}} \left\| \tilde{f}_i^{(1)} - f_i^{(1)} \right\| \tag{20}$$

$$< \sum_{i=1}^{N_{\max}} \frac{\epsilon}{2N_{\max}} + \sum_{i=1}^{N_{\max}} \frac{\epsilon}{2N_{\max}} \tag{21}$$

$$= \epsilon. \tag{22}$$

In total, this means that $\|h_\theta - R\|_{L_1} < \epsilon$, so we are done. □

Proposition B.4 (GSFs capture known baselines)**.** *Let $\mathcal{B}$ be the set of scoring functions implemented by the Min/Max/Mean aggregated probability methods (Guerreiro et al., 2022; Kadavath et al., 2022; Varshney et al., 2023; Huang et al., 2023b) for HD, as well as the MinK% (Shi et al., 2023) and MinK%++ (Zhang et al., 2024) methods for DCD. For any scoring function $f \in \mathcal{B}$, there exists a choice of functions κ, T, g such that the GSF R in Equation* (6)*, implements f.*

Proof. We will prove the Proposition by defining, for each baseline, the functions implementing components κ, T, g, assuming no ties in the ATP values $\mathbf{p}$.

Mean Aggregated Probability. This baseline simply outputs the mean across the ATPs $\mathbf{p}$. The following selection of functions implements it as a GFS:

$$\kappa(\mathbf{X}', \mathbf{p}) = \mathbf{1} \quad T(\mathbf{X}', \mathbf{p}) = 0 \quad g(\mathbf{X}', \mathbf{p}) = \frac{1}{N}\mathbf{p}$$

Min Aggregated Probability outputs the min value across the ATPs $\mathbf{p}$. The following selection of functions implements it as a GFS:

$$\kappa(\mathbf{X}', \mathbf{p}) = -\mathbf{p} \quad T(\mathbf{X}', \mathbf{p}) = -\min(\mathbf{p}) \quad g(\mathbf{X}', \mathbf{p}) = \mathbf{p}$$

Max Aggregated Probability outputs the max value across the ATPs $\mathbf{p}$. We simply pick:

$$\kappa(\mathbf{X}', \mathbf{p}) = \mathbf{p} \quad T(\mathbf{X}', \mathbf{p}) = \max(\mathbf{p}) \quad g(\mathbf{X}', \mathbf{p}) = \mathbf{p}$$

MinK%. Please refer to Section 4.

MinK%++. Let $\bar{\mathbf{p}} = \frac{\log(\mathbf{p}) - \boldsymbol{\mu}}{\boldsymbol{\sigma}}$, be the normalized version of $\mathbf{p}$, with:

$$\boldsymbol{\mu}_i = \mathbb{E}_{\mathbf{X}_i'}[\log(\mathbf{X}_i')] = \sum_{v=1}^{V} \mathbf{X}_{i,v}' \cdot \log(\mathbf{X}_{i,v}'),$$

$$\boldsymbol{\sigma}_i = \sqrt{\mathbb{E}_{\mathbf{X}_i'}[(\log(\mathbf{X}_i') - \boldsymbol{\mu}_i)^2]} = \sqrt{\sum_{v=1}^{V} \mathbf{X}_{i,v}' \cdot \left(\log(\mathbf{X}_{i,v}') - \boldsymbol{\mu}_i \right)^2}, \tag{23}$$

Where $\mathbf{X}'$ is given from Equation (1).

The baseline is implemented by setting:

$$T(\mathbf{X}', \mathbf{p}) = -\text{perc}(\bar{\mathbf{p}}, K) = -\big(\text{sort}(\bar{\mathbf{p}})_{\lceil \frac{K}{100} \cdot N \rceil}\big),$$
$$\kappa(\mathbf{X}', \mathbf{p}) = -\bar{\mathbf{p}}, \quad g(\mathbf{X}', \mathbf{p}) = \frac{\bar{\mathbf{p}}}{\lceil \frac{K}{100} \cdot N \rceil}.$$

Loss as a Privacy Proxy Yeom et al. (2018). This method uses the model's negated loss as a proxy for contamination, which can be defined as the average of the log ATPs. The method can thus be implemented with:

$$\kappa(\mathbf{X}', \mathbf{p}) = \mathbf{1}, \quad T(\mathbf{X}', \mathbf{p}) = 0, \quad g(\mathbf{X}', \mathbf{p}) = \frac{1}{N} \log(\mathbf{p}). \tag{24}$$

□

Corollary B.5 (Approximation of Baselines by LOS-NET)**.** *Our architecture, as defined in Equation* (3)*, can arbitrarily well approximate, in the L_1 sense, any of the baseline methods in $\mathcal{B}$ when operating on context and token-vocabulary of, resp., maximal sizes N_{max} and V_{max}.*

Proof. To prove Corollary 4.3, it suffices to show the following. First (i), that the baselines can be implemented as in Equation (6), given their sequence length and vocabulary size satisfy, $N \leq N_{\text{max}}, \quad V \leq V_{\text{max}}$, where values in the inputs for indices larger than N, V are 'padded' with e.g., -1. Second (ii), that their implementations are realized with κ, T, and g which are all measurable, and with g also integrable.

(i) Let us slightly modify the implementations provided in the Proof for Proposition 4.1 to correctly account for padding values. Let us conveniently define:

$$\alpha : \mathbb{R} \to \mathbb{R}, \quad \alpha(x) = 1 - \text{ReLU}(-x) = \begin{cases} 1 & x \geq 0 \\ 1 + x & x < 0 \end{cases}$$
$$N_{\text{eff}} = \sum_{i=1}^{N_{\text{max}}} \alpha(\mathbf{p}_i) \quad V_{\text{eff}} = \sum_{v=1}^{V_{\text{max}}} \alpha(\mathbf{X}_{1,v}) \tag{25}$$

as well as the following function, which will help us 'manipulate' the padding value in order not to interfere with the effective computations required by baselines:

$$\beta : \mathbb{R} \to \mathbb{R}, \quad \beta(x; M, f) = \begin{cases} f(x) & x \geq 0 \\ M & x = -1 \end{cases}, M > 0. \tag{26}$$

Mean Aggregated Probability.

$$\kappa(\mathbf{X}', \mathbf{p}) = \mathbf{1} \quad T(\mathbf{X}', \mathbf{p}) = 0 \quad g(\mathbf{X}', \mathbf{p}) = \frac{1}{N_{\text{eff}}} \mathbf{p} \circ \alpha(\mathbf{p}),$$

where $\circ$ denotes the hadamard (element-wise) product.

Min Aggregated Probability.

$$\kappa(\mathbf{X}', \mathbf{p}) = -\beta(\mathbf{p}) \quad T(\mathbf{X}', \mathbf{p}) = -\min(\beta(\mathbf{p})) \quad g(\mathbf{X}', \mathbf{p}) = \mathbf{p} \quad M = 2, f \equiv \text{id}.$$

Max Aggregated Probability.

$$\kappa(\mathbf{X}', \mathbf{p}) = \mathbf{p} \quad T(\mathbf{X}', \mathbf{p}) = \max(\mathbf{p}) \quad g(\mathbf{X}', \mathbf{p}) = \mathbf{p}$$

MinK%.

$$\kappa(\mathbf{X}', \mathbf{p}) = -\beta(\mathbf{p}) \quad T(\mathbf{X}', \mathbf{p}) = -\big(\text{sort}(\beta(\mathbf{p}))_{\lceil \frac{K}{100} \cdot N_{\text{eff}} \rceil}\big) \quad g(\mathbf{X}', \mathbf{p}) = \frac{\log(\beta(\mathbf{p}))}{\lceil \frac{K}{100} \cdot N_{\text{eff}} \rceil} \quad M = 2, f \equiv \text{id}.$$

where the note the application of β inside the $\log$ prevents negative inputs.

MinK%++. Before illustrating how this baseline is implemented, we note the following. In order for the normalization of log-probs to be well-defined, it is required that: (1) μ is finite, (2) the denominator is greater than 0. As for (1), we note that null probability values ($X_{i,v} = 0$) would be problematic, as they would cause the log function to output $-\infty$. We assume, in this case, that all probability values lie in $[\epsilon_1, 1]$, with ϵ_1 being a small value such that $0 < \epsilon_1 < 1$. Regarding (2), we see that the problematic situation would occur in cases where the probability distribution is uniform. We assume to handle this case by adding a small positive constant $\epsilon_2 > 0$ in the denominator, so that the normalization would take the form: $\bar{\mathbf{p}} = \frac{\log(\mathbf{p}) - \boldsymbol{\mu}}{\boldsymbol{\sigma} + \epsilon_2}$.

Under these assumptions, we define the following β functions:

$$\beta_1 = \beta(\cdot; 2, \text{id.}) \quad \beta_2^i = \beta(\cdot; -\frac{2\log(\epsilon_1)}{\epsilon_2}, f_i), \;\; f_i(x) = \frac{\log(x) - \boldsymbol{\mu}_i}{\left\lceil \frac{K}{100} \cdot N_{\text{eff}} \right\rceil \boldsymbol{\sigma}_i + \epsilon_2}$$

where we note that $-\frac{2\log(\epsilon_1)}{\epsilon_2}$ upper-bounds all the possible values that can be attained by f_i's under our assumptions.

At this point, we observe that the values $\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i$ can be correctly obtained as follows, in a way that is not influenced by our padding scheme:

$$\boldsymbol{\mu_i} = \sum_v \alpha(\mathbf{X}'_{i,v}) \cdot \mathbf{X}'_{i,v} \log\left(\beta_1(\mathbf{X}'_{iv})\right) \tag{27}$$

$$\boldsymbol{\sigma_i} = \sqrt{\sum_v \alpha(\mathbf{X}'_{i,v}) \cdot \mathbf{X}'_{i,v} \left(\log(\beta_1(\mathbf{X}')_{i,v}) - \boldsymbol{\mu}_i\right)^2} \tag{28}$$

At this point, let $\boldsymbol{\beta}_2(\mathbf{p})_i = \beta_2^i(\mathbf{p}_i)$. We set:

$$\kappa(\mathbf{X}', \mathbf{p}) = -\boldsymbol{\beta}_2(\mathbf{p}) \quad T(\mathbf{X}', \mathbf{p}) = -\big(\text{sort}(\boldsymbol{\beta}_2(\mathbf{p}))_{\left\lceil \frac{K}{100} \cdot N_{\text{eff}} \right\rceil}\big) \quad g(\mathbf{X}', \mathbf{p}) = \frac{\boldsymbol{\beta}_2(\mathbf{p})}{\left\lceil \frac{K}{100} \cdot N_{\text{eff}} \right\rceil}$$

and note that the K-th percentile in T is correctly computed despite the padding values due to the specific choice of M in β_2's.

Loss as a Privacy Proxy Yeom et al. (2018).

$$\kappa(\mathbf{X}', \mathbf{p}) = \mathbf{1}, \quad T(\mathbf{X}', \mathbf{p}) = 0, \quad g(\mathbf{X}', \mathbf{p}) = \frac{1}{N_{\text{eff}}} \log(\mathbf{p}). \tag{29}$$

(ii) We now proceed to show that the implementations above are obtained via measurable functions κ, T, and a measurable and integrable function g, which completes the proof.

Step 1: Consider a fixed sequence length $N' \in [N_{\text{max}}]$ and a fixed vocabulary size $V \in [V_{\text{max}}]$. When restricted to these parameters, all relevant functions are continuous. This follows from the fact that each function, when restricted in this manner, is composed of continuous functions.

Step 2: The input domain for each combination of sequence length $N' \in [N_{\text{max}}]$ and vocabulary size $V \in [V_{\text{max}}]$ forms a compact set, and the union of all of this domains is also compact (as a finite union of compact sets). Moreover, for any two distinct pairs (N_1, V_1) and (N_2, V_2), if either $N_1 \neq N_2$ or $V_1 \neq V_2$, then the corresponding domains are disjoint.

In most of our cases of interest, this follows from the fact that probabilities lie within $[0, 1]$ and that padding is represented by -1. In other cases, e.g., the application of β, the sets might be different, but remain disjoint and compact.

Thus, by the following lemma, all functions κ, T, g for all baselines are continuous, completing the proof.

Lemma B.6. *Let X be a subset of a metric space, which is compact, and can be expressed as a finite disjoint union of compact subsets X_i indexed by a finite set I, i.e.,*

$$X = \bigsqcup_{i \in I} X_i.$$

Suppose a function $f : X \to \mathbb{R}^n$ is defined such that for each $i \in I$, there is a continuous function

$$g^{(i)} : X_i \to \mathbb{R}^n$$

satisfying $f|_{X_i} = g^{(i)}$. Then, f is continuous on X.

The finite disjoint union of compact subsets correspond to all possible sequence lengths ($N' \in N_{\text{max}}$) and vocabulary sizes ($V' \in V_{\text{max}}$). Below we provide the proof for Lemma B.6.

Proof. Consider any point $\mathbf{x} \in X$, and let $(\mathbf{x}^{(m)})$ be a sequence converging to $\mathbf{x}$, in X. We need to show that

$$f(\mathbf{x}^{(m)}) \to f(\mathbf{x}) \quad \text{as } m \to \infty.$$

Since X is a finite disjoint union of compact subsets X_i, there exists an index i^* such that $\mathbf{x} \in X_{i^*}$.

Since the subsets X_i are disjoint and compact, there exists a positive minimum separation distance between distinct subsets, defined as,

$$\delta^* = \frac{1}{2} \min_{i \neq j} \inf_{\mathbf{x} \in X_i, \mathbf{y} \in X_j} \|\mathbf{x} - \mathbf{y}\|.$$

Since each X_i is compact and the index set is finite[7], this minimum distance is well-defined and strictly positive.

Because $\mathbf{x}^{(m)} \to \mathbf{x}$, there exists an integer M such that for all $m > M$, we have

$$\|\mathbf{x}^{(m)} - \mathbf{x}\| < \delta^*.$$

By the definition of δ^*, this ensures that for sufficiently large m, the sequence $\mathbf{x}^{(m)}$ remains in X_{i^*}, i.e., $\mathbf{x}^{(m)} \in X_{i^*}$ for all $m > M$.

Since f coincides with $g^{(i^*)}$ on X_{i^*}, we have

$$f(\mathbf{x}^{(m)}) = g^{(i^*)}(\mathbf{x}^{(m)}), \quad \text{for all } m > M.$$

By assumption, $g^{(i^*)}$ is continuous on X_{i^*}, so

$$g^{(i^*)}(\mathbf{x}^{(m)}) \to g^{(i^*)}(\mathbf{x}) \quad \text{as } m \to \infty.$$

Since $f(\mathbf{x}) = g^{(i^*)}(\mathbf{x})$, it follows that

$$f(\mathbf{x}^{(m)}) \to f(\mathbf{x}),$$

which proves that f is continuous at $\mathbf{x}$. Since $\mathbf{x}$ was arbitrary, f is continuous on X. □

□

C Extended Experimental Section

C.1 Experimental Details

Our experiments were conducted using the PyTorch Paszke et al. (2019) framework, using NVIDIA L40 GPUs. We use a fixed batch size of 64 for all the tasks and datasets, and a fixed value of 8 heads (except for the MoviesOrgad et al. (2024) dataset) in our light-weight transformer encoder for LOS-Net. Hyperparameter tuning was performed utilizing the Weight and Biases framework Biewald (2020) – see Table 4.

C.2 HyperParameters

In this section, we detail the hyperparameter search conducted for our experiments. We use the same hyperparameter grid for our main model, LOS-Net, and our proposed learning-based baselines, namely, ATP+R-MLP, ATP+R-Transf.. Additionally, we note that for a given dataset, we maintained the same grid search approach for all LLMs' LOSs that we have trained on. The hyperparameter search configurations for all datasets are presented in Table 4. The grid search optimizes for the AUC calculated on the validation set.

[7] https://proofwiki.org/wiki/Distance_between_Disjoint_Compact_Set_and_Closed_Set_in_Metric_Space_is_Positive#google_vignette

Table 4: Hyperparameter search grid for LOS-NET.

Dataset	Num. layers	Learning rate	Embedding size	Epochs	Dropout	Weight Decay
HOTPOTQA	$\{1, 2\}$	$\{0.0001\}$	$\{128, 256\}$	$\{300\}$	$\{0, 0.3\}$	$\{0, 0.001\}$
IMDB	$\{1, 2\}$	$\{0.0001\}$	$\{128, 256\}$	$\{300\}$	$\{0, 0.3\}$	$\{0, 0.001\}$
MOVIES	$\{1, 2\}$	$\{0.0001\}$	$\{128, 256\}$	$\{300, 500\}$	$\{0.0, 0.3, 0.5\}$	$\{0, , 0.001, 0.005\}$
WIKIMIA (32/64)	$\{1, 2\}$	$\{0.0001\}$	$\{128, 256\}$	$\{100, 500, 1000\}$	$\{0, 0.3\}$	$\{0, 0.001\}$
BOOKMIA	$\{1, 2\}$	$\{0.0001\}$	$\{64, 128\}$	$\{500\}$	$\{0, 0.3, 0.5\}$	$\{0, 0.001\}$

C.3 OPTIMIZERS AND SCHEDULERS

For all datasets we employ the AdamW optimizer Loshchilov (2017) paired with a Linear scheduler, using a warm up of 10% of the epochs. We apply an early stopping criterion if there is no improvement in validation performance for 30 consecutive epochs.

C.4 OUR BASELINES AND RANK ENCODING

ATP+R-Transf. This baseline is implemented as described in Equation (3), but without incorporating the TDS ($\mathbf{X}$), as follows:

$$h_\theta(\mathbf{X}, \mathbf{p}) = \mathcal{T}\left(\text{RE}(\mathbf{X}, \mathbf{p})\right), \tag{30}$$

where $\mathcal{T}$ represents an encoder-only transformer architecture Vaswani (2017).

ATP+R-MLP. This baseline is similar to **ATP+R-Transf.** but replaces the transformer with an MLP. Formally:

$$h_\theta(\mathbf{X}, \mathbf{p}) = \text{MLP}\left(\text{RE}(\mathbf{X}, \mathbf{p})\right), \tag{31}$$

C.5 DATASET DESCRIPTION

C.5.1 DATASETS FOR HALLUCINATION DETECTION

In this section, we provide an overview of the three datasets used in our hallucination detection analysis; we mostly follow the framework given in Orgad et al. (2024) in constructing the datasets. Our aim was to ensure coverage of a wide variety of tasks, required reasoning skills, and dataset diversity. For each dataset, we highlight its unique contributions and how it complements the others.

For all datasets, we used a consistent split of 10,000 training samples and 10,000 test samples.

1. **HotpotQA** Yang et al. (2018): This dataset is specifically designed for multi-hop question answering and includes diverse questions that require reasoning across multiple pieces of information. Each entry comprises supporting Wikipedia documents that aid in answering the questions. For our analysis, we utilized the "without context" setting, where questions are posed directly. This setup demands both factual knowledge and reasoning skills to generate accurate answers.
2. **Movies** Orgad et al. (2024): To evaluate generalization in scenarios regarding movies involving factual inaccuracies (i.e., hallucinations), we employed the dataset introduced by Orgad et al. (2024).
3. **IMDB** Maas et al. (2011): This dataset contains movie reviews designed for sentiment classification tasks. Following the approach outlined in Orgad et al. (2024), we applied a one-shot prompt to guide the large language model (LLM) in using the predefined sentiment labels effectively.

Annotation collection for HD. Specifically, following Orgad et al. (2024), the dataset $D = \{(q_i, z_i)\}_{i=1}^{\ell}$ contains ℓ question-answer pairs, where q_i are questions and z_i are ground-truth answers. For each q_i, the model generates a response $\hat{z}_i$, with predicted answers $\{\hat{z}_i\}_{i=1}^{\ell}$. The LOS for each response, $\{(\mathbf{X}, \mathbf{p})_i\}_{i=1}^{\ell}$, is saved. Correctness labels $y_i \in \{0, 1\}$ are assigned by comparing $\hat{z}_i$ to z_i, resulting in the error-detection dataset $\{(\mathbf{X}, \mathbf{p})_i, y_i\}_{i=1}^{\ell}$.

C.5.2 DATASETS FOR DATA CONTAMINATION DETECTION

BookMIA. The original BookMIA data have been obtained from the Hugging Face dataset `swj0419/BookMIA`[8], accessed via the Hugging Face python `datasets` API. The dataset totals $9,870$ excerpts from a total of 100 books, of which 50 are labeled as members (positives) and 50 are labeled as non-members (negatives).

Throughout all experiments on BookMIA, including the evaluation of baselines, we process only the first 128 words from each excerpt, originally 512-word long. This expedient allowed for faster LLM inference and lighter data storage at the time of dataset creation, i.e., the extraction and saving of LLM outputs.

As no standard split is available for this dataset, we proceed by randomly forming *training* and *test* sets in the proportions of, resp., 80% and 20%. To ensure that *all* excerpts from the same book are in either one of the two sets (and never in both), we first separate books into two separate lists based on their label, shuffle the obtained lists using a random seed of 42, and then, for each of the two lists, take the first 80% of books as training books, and the remaining 20% as test books. Training and test sets are obtained by taking the corresponding excerpts from, respectively, training and test books. After this, we verified that the obtained sets are both approximately class-balanced ($\approx 50\%$ of excerpts in both the training and test sets are labeled as positives).

In the case of the reference-based baseline, we consider the smallest-sized available counterparts for the respectively attacked LLMs, namely: Pythia 70M for Pythia models and Llama-1 7B for Llama models. All LLMs are accessed through the Hugging Face python interface, specifically: `EleutherAI/pythia-70m`, `EleutherAI/pythia-{6.9,12}b`[9] and `huggyllama/llama-{7,13,30}b`[10].

WikiMIA. WikiMIAShi et al. (2023) is the first benchmark for pre-training data detection, comprising texts from Wikipedia events. The distinction between training and non-training data is determined by timestamps. WikiMIA organizes data into splits based on sentence length, enabling fine-grained evaluation. It also considers two settings: original and paraphrased. The original setting evaluates the detection of verbatim training texts, while the paraphrased setting, where training texts are rewritten using ChatGPT, assesses detection on paraphrased inputs. In this paper, we consider the original (non-paraphrased) split and focus on the 32 and 64 split sizes, as they contain the largest number of samples, approximately 750 and 550, respectively.

C.6 ABLATION STUDY

Existing methods often overlook a critical aspect of LOS. Specifically, they primarily rely on the ATP, $\mathbf{p}$, while neglecting the TDS, $\mathbf{X}$. In this subsection, we conduct an ablation study to evaluate the significance of the TDS in general, as well as its size, namely the hyperparameter K introduced in Equation (1).

The Role of the TDS ($\mathbf{X}$). As a case study, we examine both the DCD task on the BookMIA dataset and the HD task across the three datasets: HotpotQA, IMDB, and Movies. Our analysis involves six LLMs. To examine to role of the TDS, we benchmark LOS-NET against our two proposed baselines, which focus on the ATP, namely, ATP+R-TRANSF. and ATP+R-MLP.

The results, summarized in Figure 5, consistently demonstrate that the best-performing model is LOS-NET. In many cases, LOS-NET outperforms the alternatives by a significant margin, indicating that the information encoded in the TDS ($\mathbf{X}$) is crucial for both tasks.

When comparing the two ATP-based baselines, ATP+R-TRANSF. and ATP+R-MLP, we find that in 8 out of 10 cases, ATP+R-TRANSF. achieves better performance. This suggests that treating ATP ($\mathbf{p}$) as sequential data - the more natural approach - is more effective than using an MLP.

[8] `https://huggingface.co/datasets/swj0419/BookMIA`.

[9] `https://huggingface.co/EleutherAI/pythia-70m`, `https://huggingface.co/EleutherAI/pythia-6.9b`, `https://huggingface.co/EleutherAI/pythia-12b`.

[10] `https://huggingface.co/huggyllama/llama-7b`, `https://huggingface.co/huggyllama/llama-13b`, `https://huggingface.co/huggyllama/llama-30b`.

The hyperparameter K. To evaluate the impact of the hyperparameter K introduced in Equation (1), we conduct a comprehensive case study focusing on the task of HD. This analysis is performed across two LLMs, Llama3-8b-instruct and Mistral-7b-instruct, as well as two diverse datasets, HotpotQA and IMDB.

We experiment with various values of K, specifically $K \in \{10, 20, 50, 100, 300, 500, 800, \text{“Full Vocabulary”}\}$, and applied the exact same hyperparameters grid search over the other hyperparameters (i.e., number of layers, learning rate) for all of those values. The corresponding results are presented in Figure 4.

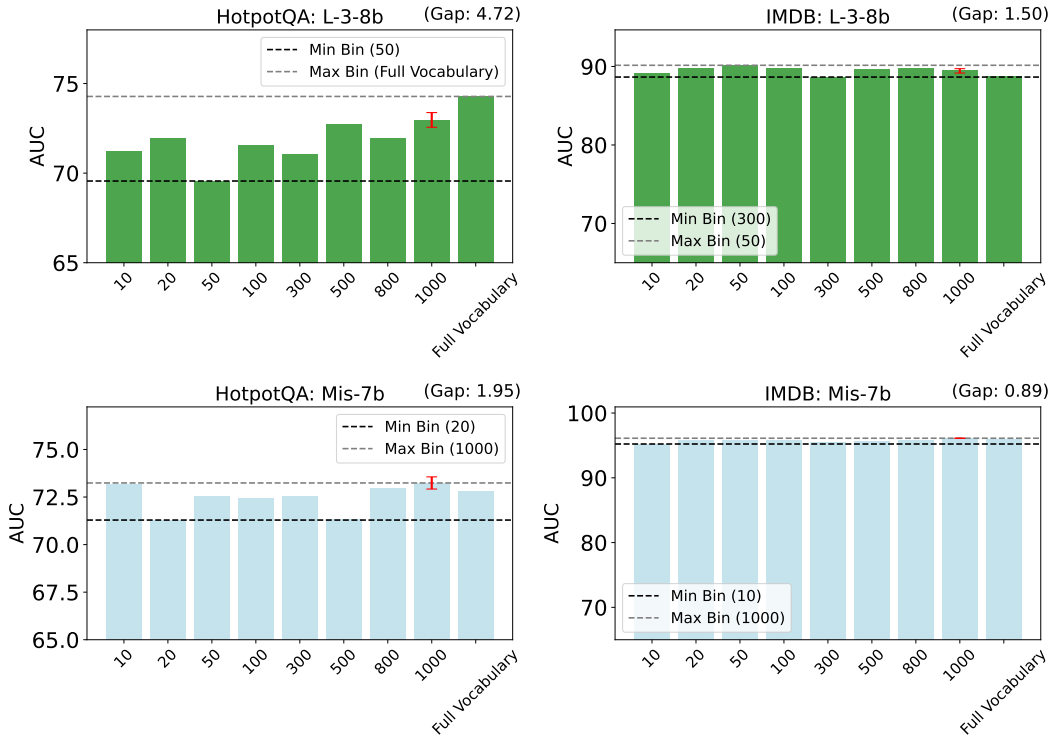

Figure 4: Ablation study analyzing the effect of the hyperparameter K introduced in Equation (1). The gap between the highest and lowest bin is indicated on the top right of each histogram.

It is evident that the hyperparameter K has little impact on the final results. Specifically, three out of the four gaps are less than 2 AUC, and the largest gap is below 5 AUC. Surprisingly, using the full vocabulary (the rightmost bin in all histograms) does not show any clear performance improvement but significantly slows down training compared to $K \leq 1000$. Thus, as an optimal balance between training time and performance, we select $K = 1000$ for all the experiments unless otherwise specified.

C.7 Extended Results for Transferability

Transfer Learning across LLMs and Datasets for HD. Differently than DCD, where zero-shot application of LOS-Net was successful, for HD we observed non-trivial generalization in the zero-shot setup, however, not sufficient to surpass the simple probability-based techniques. This led us to investigate LOS-Net capabilities in a transfer learning setting, in which we conduct a rapid fine-tuning procedure on all possible LLM/datasets combinations. Specifically, we perform a 10-epoch fine-tuning on the target LLM/dataset (as opposed to 300+ epochs in our standard setting). This process was measured to take less than a minute. We benchmark the fine-tuned model against two baselines. First, to test for successful transfer, we compare with a LOS-Net trained from scratch under an identical setup (i.e., 10 epochs). Second, we contrast the fine-tuned model with the best-

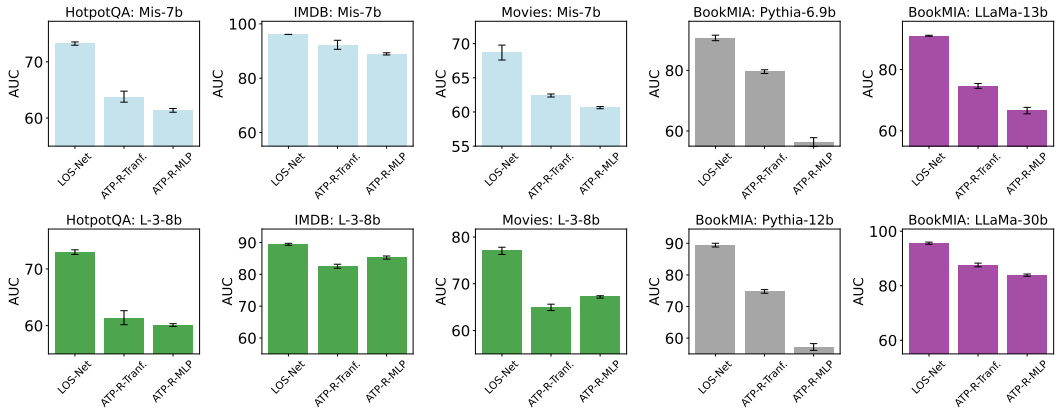

Figure 5: Ablation study evaluating the role of the TDS ($\mathbf{X}$) and the ATP ($\mathbf{p}$). Results are shown for the four LLMs: L-3-8b, Pythia-12b, Mis-7b, and LLaMa-13b.

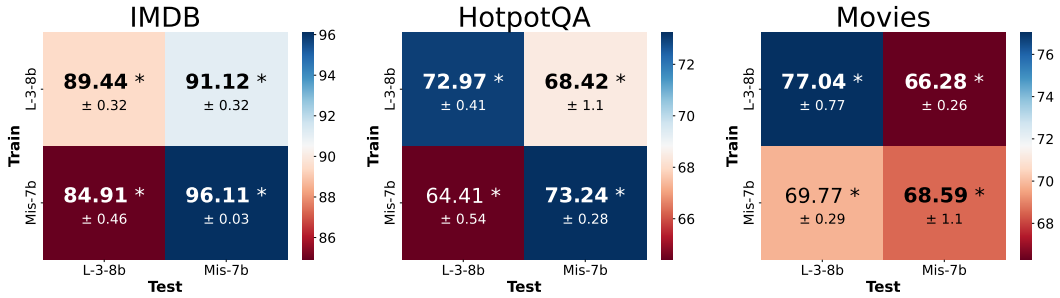

Figure 6: Cross-LLM transfer learning performance (AUC). Y-axis: source LLMs (train), X-axis: target LLMs (test). Results better than the best baseline are in **Bold**; results surpassing training from scratch are marked with *.

reported baseline; this is crucial, as generalization scores above 0.5 AUC are only useful if they surpass non-learnable baselines relying solely on probas/logit outputs.

The test AUC of our fine-tuned LOS-NET's are reported in Figures 6 and 7. A superscript (*) indicates that the fine-tuned LOS-NET achieves better performance compared to training from scratch. Bold text highlights cases where the fine-tuned LOS-NET outperforms the best (non-learnable) baseline methods.

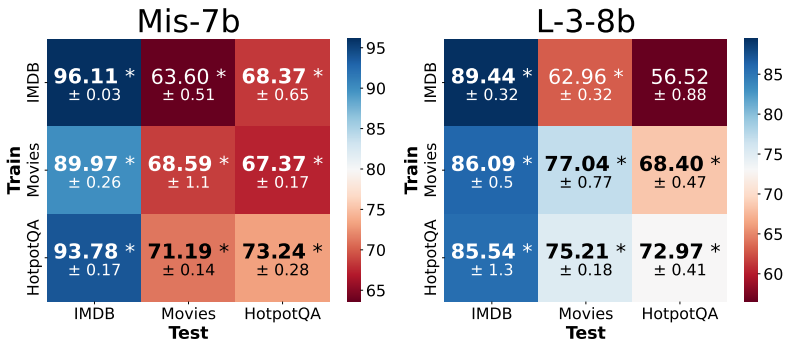

Figure 7: Cross-dataset transfer learning performance (AUC). Y-axis: train datasets, X-axis: test datasets. Results better than the best baseline are in **Bold**; results surpassing training from scratch are marked with *.

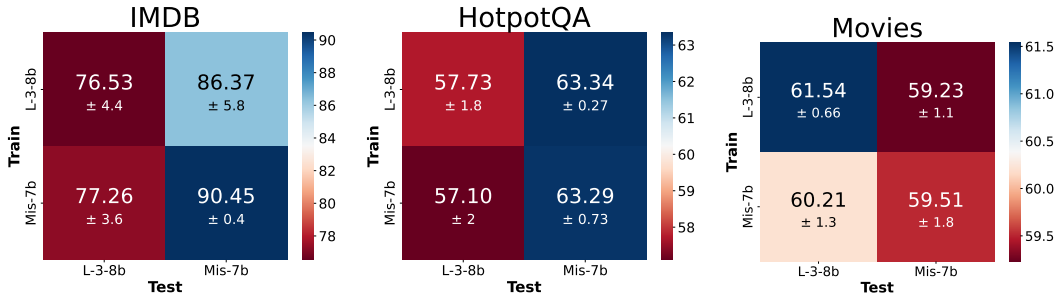

Figure 8: Training from scratch performance (AUC) in the Cross-LLM setting. Y-axis: source LLMs (train), X-axis: target LLMs (test).

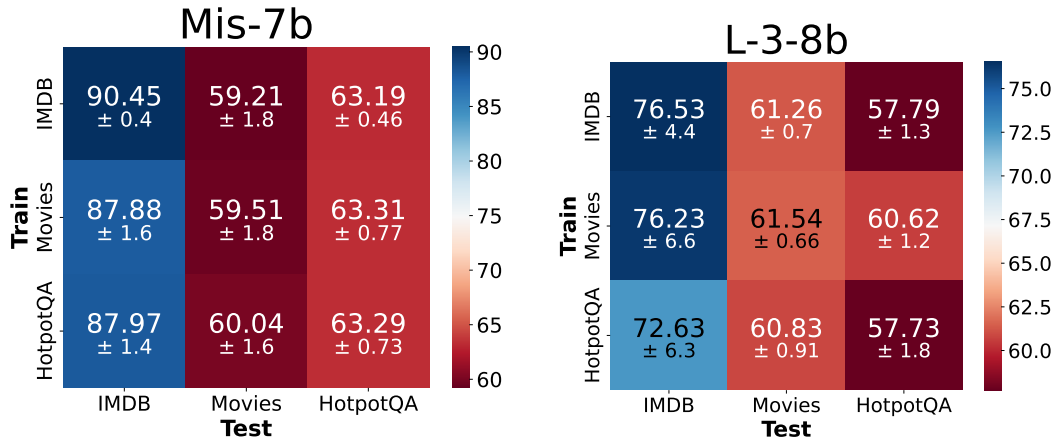

Figure 9: Training from scratch performance (AUC) in the cross-dataset setting. Y-axis: train datasets, X-axis: test datasets.

We also present the complementary results for Figures 6 and 7, for the case where we trained from scratch. The results are presented in Figures 8 and 9.

D Additional Tasks Background

In this section, we provide some additional background and motivation for the DCD and HD tasks.

Data Contamination Detection. Large-scale pre-training of LLMs typically involves crawling vast amounts of online data, a common practice to meet their substantial data requirements. However, this approach risks exposing models to evaluation datasets, potentially compromising our ability to assess their generalization performance accurately Brown et al. (2020), or, taking a different perspective, can pose legal and ethical issues when models are accidentally exposed to copyrighted or sensitive data during training. This phenomenon is typically referred to as Data Contamination. Recently, Li et al. (2024b) demonstrated that LLMs from the widely used LLaMA Touvron et al. (2023) and Mistral Jiang et al. (2023) model families exhibit significant data contamination, particularly concerning frequently used evaluation datasets.

Hallucination Detection. LLMs' tendency to generate untrustworthy outputs, commonly known as "hallucinations," remains a significant challenge to their widespread adoption in real-world applications Tonmoy et al. (2024). To address this issue, various hallucination mitigation techniques have been proposed, including retrieval-augmented generation Lewis et al. (2020); Izacard et al. (2023); Gao et al. (2023), customized fine-tuning Maynez et al. (2020); Cao et al. (2022); Qiu et al. (2023), and, inference-time manipulation Li et al. (2024a); Qiu et al. (2024); Zhao et al. (2024), to name a few. However, applying these methods to all user-LLM interactions can be computationally ex-

pensive. As a more targeted approach, hallucination detection has been explored to enable selective intervention only when necessary.

General Considerations on Annotations. We consider access to a set of annotations y's, which we naturally associate with the corresponding LOS elements. These encode ground-truth labels pertaining to problems of interest, e.g., whether the input text $\vec{s}$ is in the pretraining corpus of f, or whether f hallucinated when generating $\vec{g}$ from prompt $\vec{s}$. Collecting these annotations is generally possible, and various strategies could be adopted. For example, for DCD, labels can be gathered with collaborative efforts testing for text memorization, as studied e.g. in (Chang et al., 2023). We also note that annotations are immediately (and trivially) available for open-source LLMs with disclosed pretraining corpora such as Pythia (Biderman et al., 2023). As we demonstrated in Section 5, models trained on annotations available for one LLM can, in some cases, be *transferred* and applied to another LLM.

For HD, ground-truth labels can be collected by providing the target LLM with inputs prompting for completion or question answering on known facts and/or reasoning tasks. Hallucinations or error annotations are derived by comparing the consistency of the model's response with known, factually true, or logically correct answers. For further details, refer to Appendix C.5.1.