Learning on LLM Output Signatures for Gray-Box Behavior Analysis

Anonymous Authors[1]

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.

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), where determining whether an LLM is fabricating information or has been exposed to specific training data is crucial for safe and reliable deployment.

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 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 using heuristics, but a principled framework to utilize this data is still lacking.

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 trans-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on ICML 2025 Workshop on Reliable and Responsible Foundation Models. Do not distribute.

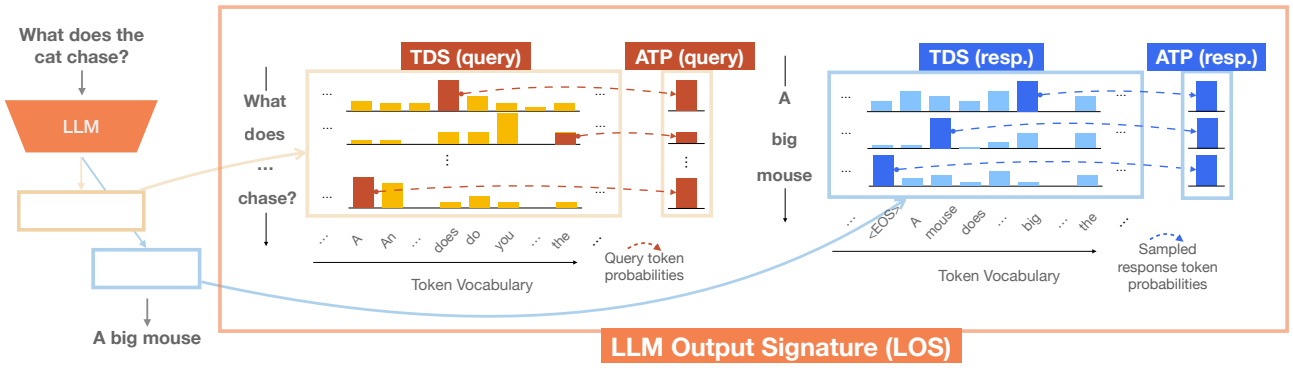

Figure 1: Left: The LLM processes the input "What does the cat chase?" and generates the output "A big mouse". Right: The corresponding query/response ("resp.") Token Distribution Sequences (TDS) and Actual Token Probabilities (ATP), together constituting the LLM Output Signature (LOS). We propose to analyze LLM behavior by learning directly from this unified representation.

former encoder[1] operating on an effective encoding of ATP, TDS, and their interactions. We prove that LOS-NET can 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). Our comprehensive empirical study on DCD and HD demonstrates a substantial information gap between using the complete LOS and relying solely on ATP. Notably, LOS-NET improves over all considered baselines across both tasks, often by a significant margin. Crucially, our architecture is extremely efficient, with detection times of $\approx 10^{-5}$s per instance, making it a compelling approach for applications such as on-line error detection for guided-generation. LOS-NET also exhibits promising dataset-level transfer and strong cross-model generalization, the latter suggesting its viable application to real-world tasks such as copyright-infringement detection over closed-source LLMs (see, e.g., our results on the BookMIA benchmark (Shi et al., 2023) in Section 5.2).

Contributions are summarized as follows: (1) we introduce LOS as a suitable representation for analyzing LLM behavior, (2) we develop an effective and a 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.

[1]Around 1M parameters.

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 C 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 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 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 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

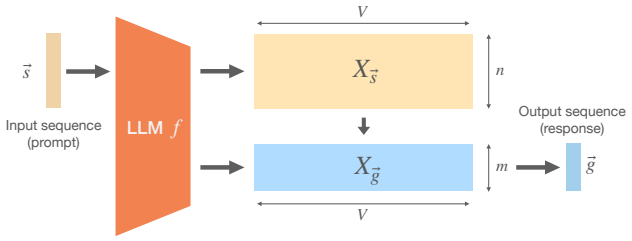

Figure 2: LLM processing pipeline. Token sequence $\vec{s}$ is processed by an LLM f, generating full TDSs $\mathbf{X}_s, \mathbf{X}_g$ for input $\vec{s}$ and response $\vec{g}$.

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. 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, $(\mathbf{p}_s)_i \coloneqq \mathbf{X}_{i,v}$ 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 may 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 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. See illustration in Appendix D, Figure 9. 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],

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

where $\odot$ is the hadamard 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 $\text{RE}(\mathbf{X}, \mathbf{p})$, we first linearly project $\mathbf{X}'$, concatenate it with $\text{RE}(\mathbf{X}, \mathbf{p})$, and then feed it to an encoder-only transformer $\mathcal{T}$ with learnable positional encodings, operating in the temporal dimension (Vaswani, 2017):

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

Here, $\mathbf{W} \in \mathbb{R}^{K \times K'}$, $\|$ denotes concatenation on the feature dimension, and θ includes all parameters, $\mathbf{w}_1, \mathbf{w}_2, \mathbf{W}$ and the parameters of $\mathcal{T}$. Finally, we pool over the [CLS] token and obtain output scores via a linear layer. The resulting model, LOS-NET, is trained with binary cross-entropy loss.

[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

Here, 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 significantly outperforming them in practice, as shown in our experiments. Proofs are enclosed in Appendix A. 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 via a family of functions (see next).

Gated Scoring Functions (GSFs). We define the family of *Gated Scoring Functions* (GSF) as the set of functions scoring 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 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: $T(\mathbf{X}', \mathbf{p}) = -\text{perc}(\mathbf{p}, K) = -\big(\text{sort}(\mathbf{p})_{\lceil \frac{K}{100} \cdot N \rceil}\big)$, $\kappa(\mathbf{X}', \mathbf{p}) = -\mathbf{p}$, $g(\mathbf{X}', \mathbf{p}) = \frac{\log \mathbf{p}}{\lceil \frac{K}{100} \cdot N \rceil}$. Refer to Appendix A for 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 (LOS-NET can approximate Equation (6))**.** *Assume maximal possible vocabulary size V_{max} and 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.*

To prove this result, we build on existing universality results on approximating continuous functions with Transformers (Yun et al., 2019), showing that our (generally non-continuous) target functions can be approximated by continuous functions. Importantly, Proposition 4.2 implies that, as long as the LOS space of interest lies within a compact domain[3], our model can approximate the general GSF in Equation (6) of LOSs for any LLM under mild conditions on κ, T, and g, potentially generalizing across LLMs (as our 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. 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:

[3]This is inherently satisfied when using probabilities; or via clamping in the case of logits or log-probs.

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 follows from Propositions 4.1 and 4.2.

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 datasets, suggesting the emergence of universal patterns in LLM behavior from the perspective of the LOS? (Section 5.3); **(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 B.6). In the following, we present our main results, and refer to Appendix B for additional experimental results and 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 B.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, where we also experiment with an additional LLM (Qwen-2.5-7b-Instruct (Yang et al., 2024)). 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. All LOS-NET experiments were conducted using the PyTorch (Paszke et al., 2019) framework on a single NVIDIA L-40 GPU.

Newly introduced learning-based baselines. In addition to task-specific baselines, we also introduce two novel learning-based baselines to appreciate the contribution of the TDS: ATP+R-MLP, ATP+R-TRANSF.. Specifically, we ablate information about the TDS and only process the ATP and rank information with an MLP or Transformer backbone. Formal definitions are in Appendix B.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, i.e., we assume no access to the LLM's internals. We assume no access to external resources – such as auxiliary LLMs (as in (Ulmer et al., 2024)), mulitple queries/generations (as in (Orgad et al., 2024; Kuhn et al., 2023)), or additional contextual cues like token-level annotations ((Orgad et al., 2024)). These approaches exhibit a significantly larger detection latency compared to LOS-NET, problematic in online applications (refer to Section 5.4 for more detailed run-time analyses and comparisons).

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), roles in Movies (Orgad et al., 2024). Details regarding the annotation process, splits and dataset sizes are in Appendix B.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), and further experiment with Qwen-2.5-7b-Instruct (Yang et al., 2024) (Q-2.5-7b).

HD Baselines. (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. These aggregations operate mean/max/min pooling 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 correctness.

Results. Table 1 presents a comprehensive summary of results on the LLMs considered in Orgad et al. (2024) (Mis-7b and L-3-8b). These 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 32 units over the best baseline for Mis-7b and 17 over the best baseline for L-3-8b. Our results further indicate that ATP learning-based baselines consistently underperform compared to LOS-NET, underscoring the critical role of the TDS, $\mathbf{X}$. Our ATP-based learnable baselines still outperform non-learnable methods in most cases, suggesting that a learning approach relying exclusively on ATP can still be a viable solution in certain scenarios. Results on Q-2.5-7b are consistent the above findings, and are deferred to Appendix B.7.

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).

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 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 B.5.2. We attack LLMs considered in (Antebi et al., 2025): LLaMa-13b/30b (Touvron et al., 2023) (L-13b/30b), Pythia-6.9b/12b(Biderman et al., 2023) (P-6.9b/12b).

DCD 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. For these baselines, we select their hyperparameters by maximizing performance on the validation set(s).

Results on BookMIA. Refer to Table 2. LOS-NET attains exceptional results, largely surpassing other reference-free approaches. Among these last ones, ours is the only

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

Table 1: Test AUCs for HD over Mis-7b and L3-8b (**bold**: best method, underlined: second best).

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

Table 2: Test AUCs on BookMIA. 'P': Pythia, 'L': LLaMa-1 (**bold**: best method, underlined: second best, pink : reference-based).

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

method that can match or outperform even the reference-LLM-based baselines. Importantly, crucial to such strong reference-free performance is to access, even partially, the TDS: our ATP-based learnable methods – which only process features for the actual sequence tokens – incur indeed significant performance drops.

Results on WikiMIA. Due to space limits, the full results on are provided in Appendix B.8, Table 5. LOS-NET consistently surpasses all baselines across all eight combinations of LLMs and datasets. The second-best method is MinK%++, followed by MinK%, consistent with the findings of (Zhang et al., 2024).

5.3. Generalization to Other LLMs and Datasets

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 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). Inspecting transfer across LLMs is, to the best of our knowledge, still unexplored. This study is important for learning-based approaches in applications such as copyright-infringement detection, where ground-truth labels may be scarce.

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.

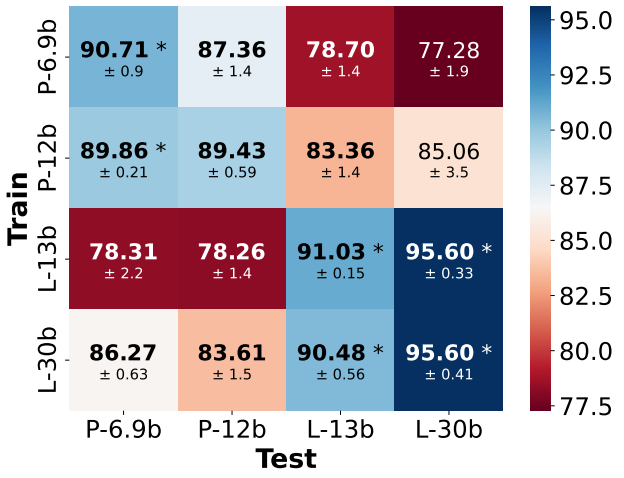

Figure 3: BookMIA zero-shot generalization (**bold**: outperforms ref-free baselines, *: outperforms ref-based ones).

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-reported non-learnable baseline. The test AUC of our fine-tuned LOS-NET's are in Figures 4 and 5. Superscript '*' indicates the fine-tuned LOS-NET is better than a counterpart trained from scratch in the same setting, **bold** indicates it outperforms the best non-learnable method.

Discussion. First, LOS-NET exhibits solid transferability in both scenarios. The finetuned models consistently outperform their counterparts trained from scratch: 16/18 cases in both the cross-LLM (Figure 4) and cross-dataset setups (Figure 5) – see '*' on the off-diagonal entries. This highlights a generally positive transfer of LOS-NET's learned representations across datasets and LLMs, and underscores the suitability of LOS as a data type in capturing generalizable patterns in LLM behaviors. Second, from a practical perspective, we find that LOS-NET outperforms the best baseline in 15/18 cases for both the cross-LLM (Figure 4) and cross-dataset (Figure 5) scenarios – see **bold** on the off-diagonal entries. Focusing on the IMDB dataset, when training on L-3-8b and testing on Mis-7b (Figure 4), our

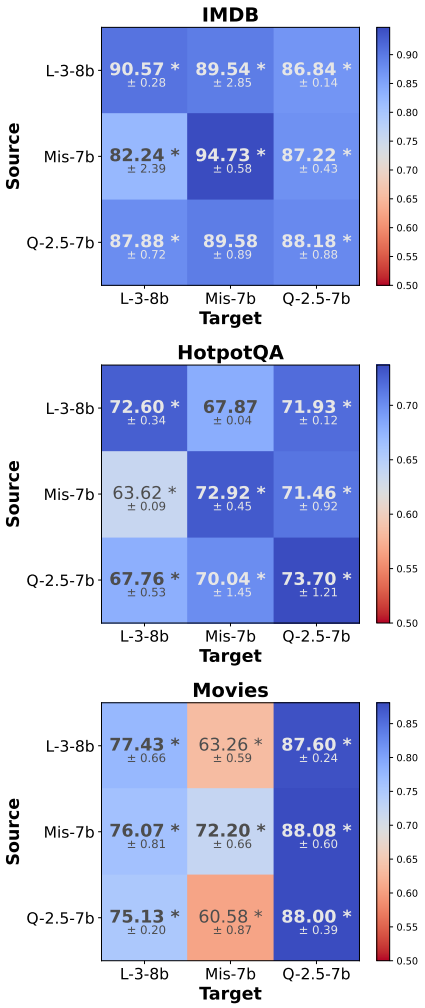

Figure 4: Cross-LLM transfer Test AUCs (cols: source LLMs, rows: target LLMs). **Bold**: finetuning LOS-NET outperforms baselines, *: it outperforms the same LOS-NET trained from scratch.

model substantially gains around 27 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 5): on Mis-7b, when training on HotpotQA or Movies and testing on IMDB, our model achieves a notable improvement of around 30 AUC units compared to the best baseline.

5.4. Run-Time Analysis

We conclude this section by discussing our comprehensive training and inference timings, reported in Appendix B.9 and Table 6. We remark how LOS-NET features an extremely contained detection latency: $\approx 10^{-5}$s per inference fwd-pass. Training is also efficient, typically completing in under one hour on a single NVIDIA L-40 GPU, and often

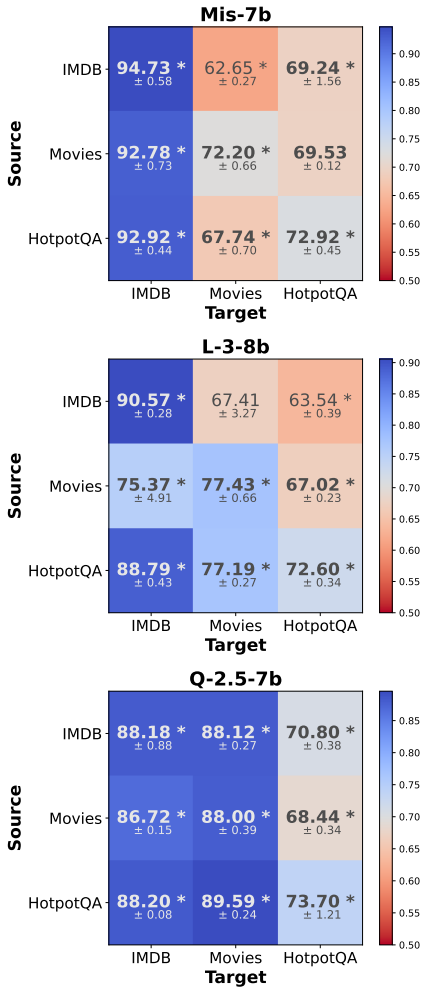

Figure 5: Cross-dataset transfer Test AUCs (cols: source data, rows: target data). **Bold**: finetuning LOS-NET outperforms baselines, *: it outperforms the same LOS-NET trained from scratch.

taking significantly shorter. To contextualize this computational efficiency w.r.t. methods relying on multiple prompting/generations (Orgad et al., 2024; Kuhn et al., 2023), we measured the generation time of Mistral-7B-Instruct. Averaging over the first 10 samples from the HotpotQA dataset, we observed an average generation time of 1.93 ± 0.18 seconds per response on a server equipped with eight NVIDIA L-40 GPUs. The methods in (Orgad et al., 2024) and (Kuhn et al., 2023) require 5 and 10 generations per detection, resp., rendering their computation overhead around six orders of magnitude higher than that of LOS-NET.

6. Conclusions

We proposed LOS-NET, an efficient method to detect data contamination and hallucinations in LLMs by leveraging their output signatures (LOS), defined as the union of Token Distribution Sequences (TDS) and Actual Token Probabilities (ATP). LOS-NET consists of a lightweight transformer with learnable rank encodings applied on the whole LOS. We proved it unifies and extends existing gray-box methods under a general framework, and experimentally showed it outperforms state-of-the-art gray-box methods across datasets and LLMs. It also exhibited strong generalization capabilities of LOS-NET, both across datasets and across LLMs. Our framework could be applied to other tasks, such as detecting LLM-generated content. Additional sources of information can also be incorporated, e.g., in the absence of latency constraints, it can be interesting to include "exact-token" flags as proposed by (Orgad et al., 2024). Last, the LOS can be extended to account for multiple prompting (Kuhn et al., 2023).

Limitations. By operating in a gray-box setting, our approach is widely applicable, but misses access to predictive information residing, e.g., in the LLM's hidden states. Additionally, we note that sorting the TDS tensor removes word alignment across the vocabulary, which may be limiting in some cases.

Impact Statement

By improving the detection of data contamination and hallucinations in Large Language Models, our contribution could foster the responsible development and use of Generative AI by enhancing transparency and trustworthiness. Applications to detecting machine-generated content could also support reducing the spread of AI-fabricated misinformation. We also note, however, that our work sheds light on predictive information contained in the output next-token probability distributions. This could lead malicious actors to develop more sophisticated defense mechanisms to MIAs or lead companies to potentially limit access to LLM outputs, thus hindering advancements in open research.

References

Antebi, S., Habler, E., Shabtai, A., and Elovici, Y. Tag&tab: Pretraining data detection in large language models using keyword-based membership inference attack. *arXiv preprint arXiv:2501.08454*, 2025.

Atzmon, Y. and Chechik, G. 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.

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

Biderman, S., Schoelkopf, H., Anthony, Q. G., Bradley,

H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., 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.

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

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (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.

Cao, M., Dong, Y., He, J., and Cheung, J. C. K. Learning with rejection for abstractive text summarization. In Goldberg, Y., Kozareva, Z., and Zhang, Y. (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/.

Carlini, N., Liu, C., Erlingsson, Ú., Kos, J., and Song, D. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX security symposium (USENIX security 19)*, pp. 267–284, 2019.

Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

Carlini, N., Chien, S., Nasr, M., Song, S., Terzis, A., and Tramer, F. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914. IEEE, 2022.

Chang, K., Cramer, M., Soni, S., and Bamman, D. Speak, memory: An archaeology of books known to ChatGPT/GPT-4. In Bouamor, H., Pino, J., and Bali, K. (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/.

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., and Wang, H. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

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

Guerreiro, N. M., Voita, E., and Martins, A. F. Looking for a needle in a haystack: A comprehensive study of hallucinations in neural machine translation. *arXiv preprint arXiv:2208.05309*, 2022.

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

Hewitt, J. and Manning, C. D. A structural probe for finding syntax in word representations. In Burstein, J., Doran, C., and Solorio, T. (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/.

Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 2023a.

Huang, Y., Song, J., Wang, Z., Zhao, S., Chen, H., Juefei-Xu, F., and Ma, L. Look before you leap: An exploratory study of uncertainty measurement for large language models. *arXiv preprint arXiv:2307.10236*, 2023b.

Izacard, G., Lewis, P., Lomeli, M., Hosseini, L., Petroni, F., Schick, T., Dwivedi-Yu, J., Joulin, A., Riedel, S., and Grave, E. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43, 2023.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Kadavath, S., Conerly, T., Askell, A., Henighan, T., Drain, D., Perez, E., Schiefer, N., Hatfield-Dodds, Z., DasSarma, N., Tran-Johnson, E., et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

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

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

Li, K., Patel, O., Viégas, F., Pfister, H., and Wattenberg, M. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36, 2024a.

Li, Y., Guo, Y., Guerin, F., and Lin, C. An open-source data contamination report for large language models. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N. (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/.

Liang, D., Gonen, H., Mao, Y., Hou, R., Goyal, N., Ghazvininejad, M., Zettlemoyer, L., and Khabsa, M. Xlm-v: Overcoming the vocabulary bottleneck in multilingual masked language models. *arXiv preprint arXiv:2301.10472*, 2023.

Liu, T., Zhang, Y., Brockett, C., Mao, Y., Sui, Z., Chen, W., and Dolan, B. A token-level reference-free hallucination detection benchmark for free-form text generation. *arXiv preprint arXiv:2104.08704*, 2021.

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

Maas, A., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. 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.

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

Mattern, J., Mireshghallah, F., Jin, Z., Schölkopf, B., Sachan, M., and Berg-Kirkpatrick, T. Membership inference attacks against language models via neighbourhood comparison. *arXiv preprint arXiv:2305.18462*, 2023.

Maynez, J., Narayan, S., Bohnet, B., and McDonald, R. On faithfulness and factuality in abstractive summarization. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. (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/.

Mosca, E., Agarwal, S., Rando Ramírez, J., and Groh, G. "that is a suspicious reaction!": Interpreting logits variation to detect NLP adversarial attacks. In Muresan, S., Nakov, P., and Villavicencio, A. (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/.

Orgad, H., Toker, M., Gekhman, Z., Reichart, R., Szpektor, I., Kotek, H., and Belinkov, Y. Llms know more than they show: On the intrinsic representation of llm hallucinations. *arXiv preprint arXiv:2410.02707*, 2024.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Qiu, Y., Ziser, Y., Korhonen, A., Ponti, E., and Cohen, S. Detecting and mitigating hallucinations in multilingual summarisation. In Bouamor, H., Pino, J., and Bali, K. (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/.

Qiu, Y., Zhao, Z., Ziser, Y., Korhonen, A., Ponti, E. M., and Cohen, S. B. Spectral editing of activations for large language model alignment. *arXiv preprint arXiv:2405.09719*, 2024.

Rateike, M., Cintas, C., Wamburu, J., Akumu, T., and Speakman, S. Weakly supervised detection of hallucinations in llm activations. *arXiv preprint arXiv:2312.02798*, 2023.

Rawte, V., Chakraborty, S., Pathak, A., Sarkar, A., Tonmoy, S., Chadha, A., Sheth, A. P., and Das, A. The troubling emergence of hallucination in large language models–an extensive definition, quantification, and prescriptive remediations. *arXiv preprint arXiv:2310.04988*, 2023.

Shi, W., Ajith, A., Xia, M., Huang, Y., Liu, D., Blevins, T., Chen, D., and Zettlemoyer, L. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2023.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pp. 3–18. IEEE, 2017.

Slobodkin, A., Goldman, O., Caciularu, A., Dagan, I., and Ravfogel, S. The curious case of hallucinatory (un) answerability: Finding truths in the hidden states of overconfident large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3607–3625, 2023.

Snyder, B., Moisescu, M., and Zafar, M. B. 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.

Tonmoy, S., Zaman, S., Jain, V., Rani, A., Rawte, V., Chadha, A., and Das, A. A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv preprint arXiv:2401.01313*, 2024.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Ulmer, D., Gubri, M., Lee, H., Yun, S., and Oh, S. J. Calibrating large language models using their generations only. *arXiv preprint arXiv:2403.05973*, 2024.

Valentin, S., Fu, J., Detommaso, G., Xu, S., Zappella, G., and Wang, B. Cost-effective hallucination detection for llms. *arXiv preprint arXiv:2407.21424*, 2024.

Varshney, N., Yao, W., Zhang, H., Chen, J., and Yu, D. A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation. *arXiv preprint arXiv:2307.03987*, 2023.

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

Verma, V., Fleisig, E., Tomlin, N., and Klein, D. Ghostbuster: Detecting text ghostwritten by large language models. In Duh, K., Gomez, H., and Bethard, S. (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/.

Wu, K., Pang, L., Shen, H., Cheng, X., and Chua, T.-S. Llmdet: A third party large language models generated text detection tool. *arXiv preprint arXiv:2305.15004*, 2023.

Yang, A., Yang, B., Hui, B., Zheng, B., Yu, B., Zhou, C., Li, C., Li, C., Liu, D., Huang, F., Dong, G., Wei, H., Lin, H., Tang, J., Wang, J., Yang, J., Tu, J., Zhang, J., Ma, J., Xu, J., Zhou, J., Bai, J., He, J., Lin, J., Dang, K., Lu, K., Chen, K., Yang, K., Li, M., Xue, M., Ni, N., Zhang, P., Wang, P., Peng, R., Men, R., Gao, R., Lin, R., Wang, S., Bai, S., Tan, S., Zhu, T., Li, T., Liu, T., Ge, W., Deng, X., Zhou, X., Ren, X., Zhang, X., Wei, X., Ren, X., Fan, Y., Yao, Y., Zhang, Y., Wan, Y., Chu, Y., Liu, Y., Cui, Z., Zhang, Z., and Fan, Z. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W. W., Salakhutdinov, R., and Manning, C. D. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

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

Yin, F., Srinivasa, J., and Chang, K.-W. Characterizing truthfulness in large language model generations with local intrinsic dimension. *arXiv preprint arXiv:2402.18048*, 2024.

Yun, C., Bhojanapalli, S., Rawat, A. S., Reddi, S. J., and Kumar, S. Are transformers universal approximators of sequence-to-sequence functions? *arXiv preprint arXiv:1912.10077*, 2019.

Zhang, J., Sun, J., Yeats, E., Ouyang, Y., Kuo, M., Zhang, J., Yang, H. F., and Li, H. Min-k%++: Improved baseline for detecting pre-training data from large language models. *arXiv preprint arXiv:2404.02936*, 2024.

Zhao, Z., Monti, E., Lehmann, J., and Assem, H. Enhancing contextual understanding in large language models through contrastive decoding. In Duh, K., Gomez, H., and Bethard, S. (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. Proofs

Proposition A.1 (LOS-NET can approximate Equation (6))**.** *Assume maximal possible vocabulary size V_{max} and 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[5] 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:

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

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 $f^{(1)} : \mathcal{D} \to \mathbb{R}^{N_{\max}}$ 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 A.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 $\|g - \tilde{g}\| = \sum_{i=1}^{N} \|g_i - \tilde{g}_i\| < \epsilon$. □

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

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 A.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.

$$\begin{aligned}
\|h_\theta - R\|_{L_1} &= \left\| \sum_{i=1}^{N_{\text{max}}} \left(h_{\theta;i}^{(1)} \right) - \sum_{i=1}^{N_{\text{max}}} f_i^{(1)} \right\|_{L_1} && (16)\\
&\leq \sum_{i=1}^{N_{\text{max}}} \left\| h_{\theta;i}^{(1)} - f_i^{(1)} \right\| && (17)\\
&= \sum_{i=1}^{N_{\text{max}}} \left\| h_{\theta;i}^{(1)} + (\tilde{f}_i^{(1)} - \tilde{f}_i^{(1)}) - f_i^{(1)} \right\| && (18)\\
&\leq \sum_{i=1}^{N_{\text{max}}} \left\| h_{\theta;i}^{(1)} - \tilde{f}_i^{(1)} \right\| + \sum_{i=1}^{N_{\text{max}}} \left\| \tilde{f}_i^{(1)} - f_i^{(1)} \right\| && (19)
\end{aligned}$$

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:

$$\begin{aligned}
&\sum_{i=1}^{N} \left\| h_{\theta;i}^{(1)} - \tilde{f}_i^{(1)} \right\| + \sum_{i=1}^{N_{\text{max}}} \left\| \tilde{f}_i^{(1)} - f_i^{(1)} \right\| && (20)\\
&< \sum_{i=1}^{N_{\text{max}}} \frac{\epsilon}{2N_{\text{max}}} + \sum_{i=1}^{N_{\text{max}}} \frac{\epsilon}{2N_{\text{max}}} && (21)\\
&= \epsilon. && (22)
\end{aligned}$$

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

Proposition A.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:

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

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

The baseline is implemented by setting:

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

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 A.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:

$$\alpha : \mathbb{R} \to \mathbb{R}, \quad \alpha(x) = 1 - \mathrm{ReLU}(-x) = \begin{cases} 1 & x \geq 0 \\ 1 + x & x < 0 \end{cases}$$

$$N_{\text{eff}} = \sum_{i=1}^{N_{\max}} \alpha(\mathbf{p}_i) \quad V_{\text{eff}} = \sum_{v=1}^{V_{\max}} \alpha(\mathbf{X}_{1,v}) \tag{25}$$

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:

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

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:

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

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 A.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 A.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[6], 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

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

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

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. □

□

B. Extended Experimental Section

B.1. Experimental Details

Our experiments were conducted using the PyTorch (Paszke et al., 2019) framework (License: BSD), using a single NVIDIA L-40 GPU for all experiments regarding LOS-NET. We use a fixed batch size of 64 for all the tasks and datasets, and a fixed value of 8 heads (except for the Movies(Orgad 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 3.

B.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 3. The grid search optimizes for the AUC calculated on the validation set.

Table 3: 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	$\{1, 2\}$	$\{0.0001\}$	$\{64, 128\}$	$\{500\}$	$\{0, 0.3, 0.5\}$	$\{0, 0.001\}$

B.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.

B.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:

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

B.5. Dataset Description

B.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) (License: CC-BY-SA-4.0): 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) (License: MIT): This dataset checks for factual accuracy in scenarios regarding movies. LLMs are asked, in particular, who was the actor/actress playing a specific role in a movie of interest. This dataset contains 7857 test samples.

3. **IMDB** (originally released with no known license by 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 $\{(\mathbf{X}, \mathbf{p})_i, y_i\}_{i=1}^{\ell}$.

LLMs. We consider the following LLMs for our experiments on HD:

1. **Mistral-7b-instruct-v0.2** (Jiang et al., 2023) (License: Apache-2.0). Referred to as Mis-7b in the main text and accessed through the Hugging Face interface at `https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3`.

2. **Llama-3-8b-Instruct** (Touvron et al., 2023) (License: Llama-3[7]). Referred to as L-3-8b in the main text and accessed through the Hugging Face interface at `https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct`.

3. **Qwen-2.5-7b-Instruct** (License: Apache-2.0): Referred to as Q-2.5-7b in the main text and accessed through the Hugging Face interface at `https://huggingface.co/Qwen/Qwen2.5-7B-Instruct`.

B.5.2. Datasets for Data Contamination Detection

BookMIA. (Shi et al., 2023) The original BookMIA data have been obtained from the Hugging Face dataset `swj0419/BookMIA`[8], accessed via the Hugging Face python `datasets` API (License: MIT). 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).

[7] `https://huggingface.co/meta-llama/Meta-Llama-3-8B/blob/main/LICENSE`
[8] `https://huggingface.co/datasets/swj0419/BookMIA`.

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] (License: Llama[11]).

WikiMIA. WikiMIA(Shi et al., 2023) (License: MIT) 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. On top of the LLMs attacked in BookMIA, here we also attack Mamba-1.4b (License: Apache-2.0), accessed via the Hugging Face interface (`https://huggingface.co/state-spaces/mamba-1.4b`).

B.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. Figures 6 and 7 report a close-up comparison between LOS-NET and our two proposed baselines, which explicitly neglect the TDS, namely, ATP+R-TRANSF. and ATP+R-MLP. These plots consistently show how 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. Regarding the two ATP-based baselines, we report that ATP+R-TRANSF. obtains better performance than ATP+R-MLP in 8 out of 12 cases, but these improvements do not seem to follow a clear pattern across LLMs and datasets. The only exception is BookMIA, on which the former architecture outperformed the latter across all the four attacked LLMs.

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.

We experiment with various values of K, specifically $K \in \{10, 50, 100, 500, 1000\}$, and trained the same selected model whose results are reported in Table 1 for $K = 1000$. The corresponding Test AUCs are presented in Figure 8.

From the reported bar plots, we do observe that performances either weakly increase with K (see, e.g., Movies for Q-2.5-7b or HotpotQA on L-3-8b), or stay approximately constant (see, e.g., IMDB on Mis-7b). In any case, the performance difference w.r.t. our default setting $K = 1000$ remains contained. This is a valuable feature, as it unlocks the effective application of LOS-NET even on non fully open LLMs such as the most recent models released by OpenAI[12].

[9] `https://huggingface.co/EleutherAI/pythia-70m` (License: Apache-2.0), `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`.

[11] `https://huggingface.co/huggyllama/llama-13b/blob/main/LICENSE`, `https://huggingface.co/huggyllama/llama-30b/blob/main/LICENSE`

[12] At the time of writing, OpenAI's API only gives access to a maximum of 20 top scoring logprobs (`https://platform.openai.`

Table 4: Test AUC scores for HD on Qwen-2.5-7b-Instruct (Q-2.5-7b). The best-performing method is in **bold**, and the second best is underlined.

Method	HotpotQA	IMDB	Movies
	Q-2.5-7b		
Logits-mean	66.2	74.8	71.3
Logits-min	59.8	72.1	42.1
Logits-max	60.4	60.7	65.1
Probas-mean	67.5	74.6	74.2
Probas-min	54.4	65.4	44.7
Probas-max	61.8	50.1	72.9
ATP+R-MLP	71.38 ± 0.28	84.69 ± 0.37	78.06 ± 0.45
ATP+R-TRANSF.	69.34 ± 2.04	87.73 ± 0.03	77.37 ± 3.13
LOS-NET	**73.71** ± 1.21	**88.19** ± 0.88	**88.00** ± 0.39

B.7. Results For Hallucination Detection for Qwen-2.5-7b

Table 4 reports results on our three considered HD datasets over LLM Qwen-2.5-7b-Instruct (Q-2.5-7b) (Yang et al., 2024). We can see LOS-NEToutperforms all non-learnable output-based baselines by large margin, as well as our learnable baselines ATP+R-TRANSF. and ATP+R-MLP.

B.8. Results On The WikiMIA Dataset

Table 5: 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

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[13] 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 5, 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 underscoring its robustness. Overall, the second-best method is MinK%++, followed by MinK%, consistent with the findings of (Zhang et al., 2024).

com/docs/api-reference/completions/create, accessed May 2025.

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

Table 6: Comparison of training and detection times of our model LOS-NET across all the DC settings explored in our paper, as well as HD settings for Mis-7b and L-3-8b. All are measured on a **single NVIDIA L-40 GPU**.

Task	Target LLM	Dataset	**Training Time** [h = hours, m = minutes, s = seconds]	**Detection Time (Mean ± Std)** [seconds]
HD	Mis-7b	HotpotQA	9m 19s	$3.32 \times 10^{-5} \pm 1.20 \times 10^{-5}$ s
		IMDB	16m 8s	$4.05 \times 10^{-5} \pm 1.83 \times 10^{-5}$ s
		Movies	17m 50s	$1.95 \times 10^{-5} \pm 7.24 \times 10^{-6}$ s
	L-3-8b	HotpotQA	6m 39s	$2.38 \times 10^{-5} \pm 7.18 \times 10^{-6}$ s
		IMDB	4m 23s	$3.37 \times 10^{-5} \pm 1.53 \times 10^{-5}$ s
		Movies	11m 34s	$3.05 \times 10^{-5} \pm 1.21 \times 10^{-5}$ s
DCD	L-13b	WikiMIA-32	33m 6s	$4.13 \times 10^{-5} \pm 1.67 \times 10^{-6}$ s
		WikiMIA-64	2m 7s	$2.67 \times 10^{-5} \pm 1.12 \times 10^{-5}$ s
		BookMIA	7m 32s	$3.67 \times 10^{-5} \pm 8.65 \times 10^{-6}$ s
	L-30b	WikiMIA-32	28m 40s	$4.05 \times 10^{-5} \pm 3.10 \times 10^{-6}$ s
		WikiMIA-64	5m 8s	$4.96 \times 10^{-5} \pm 2.54 \times 10^{-5}$ s
		BookMIA	16m 38s	$3.94 \times 10^{-5} \pm 1.42 \times 10^{-5}$ s
	P-6.9	WikiMIA-32	24m 55s	$2.91 \times 10^{-5} \pm 4.41 \times 10^{-6}$ s
		WikiMIA-64	26m 13s	$3.18 \times 10^{-5} \pm 1.56 \times 10^{-5}$ s
		BookMIA	18m 23s	$2.86 \times 10^{-5} \pm 6.16 \times 10^{-6}$ s
	P-12b	BookMIA	19m 49s	$4.07 \times 10^{-5} \pm 4.89 \times 10^{-6}$ s
	M-1.4b	WikiMIA-32	1h 6m 18s	$3.87 \times 10^{-5} \pm 1.27 \times 10^{-6}$ s
		WikiMIA-64	1h 16m 51s	$3.12 \times 10^{-5} \pm 1.42 \times 10^{-5}$ s

B.9. LOS-Net Run-Time

In Table 6, we report the wall-clock training times (for the best selected model based on the held-out validation set) and single-example detection times for LOS-NET for all experiments presented in this paper.

C. 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 expensive. 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 B.5.1.

D. LOS-Net Visualization

In Figure 9 we provide a visualization of our architecture, LOS-NET .

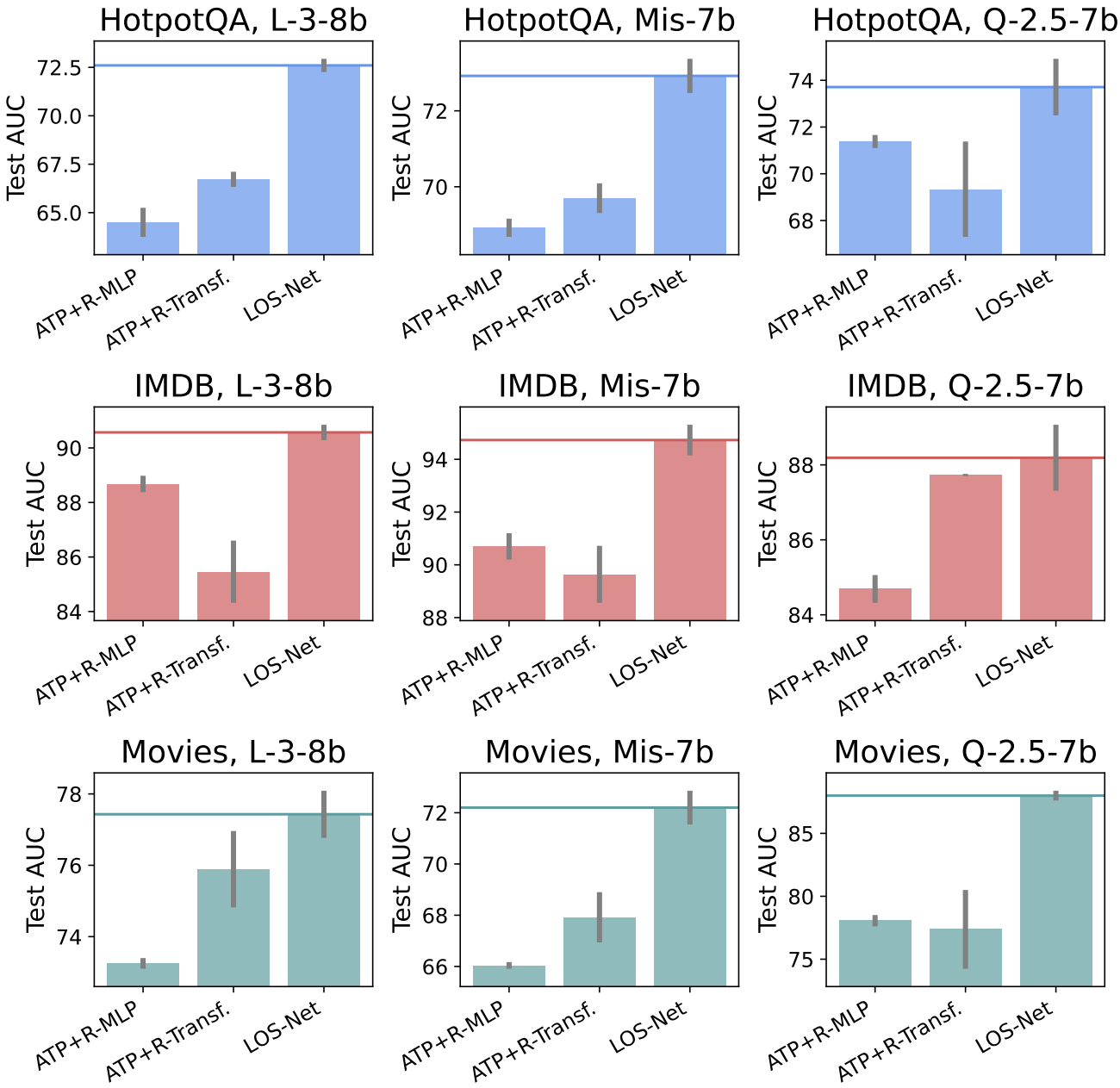

Figure 6: Ablation study evaluating the role of the TDS ($\mathbf{X}$) and the ATP ($\mathbf{p}$) on our HD setups, including datasets HotpotQA, IMDB, Movies, and LLMs L-3-8b, Mis-7b, Q-2.5-7b.

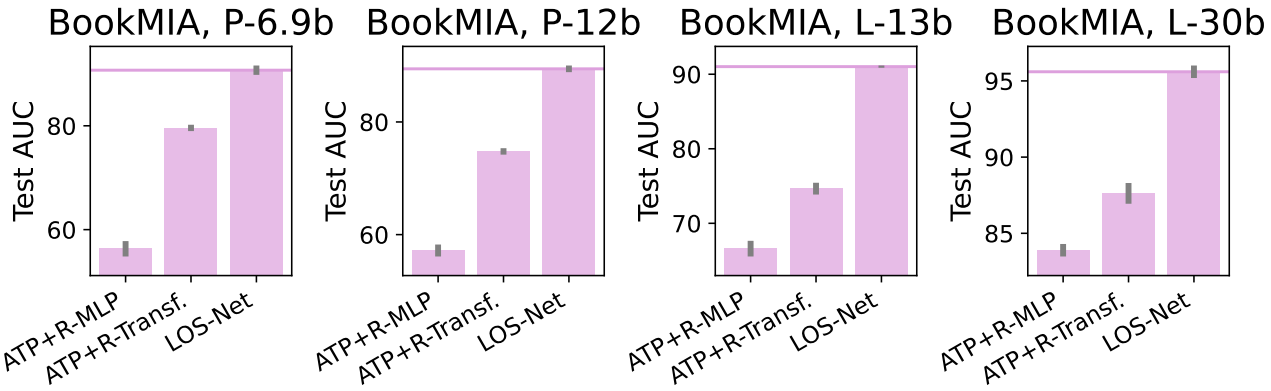

Figure 7: Ablation study evaluating the role of the TDS ($\mathbf{X}$) and the ATP ($\mathbf{p}$) on BookMIA for Pythia and Llama-1 LLMs.

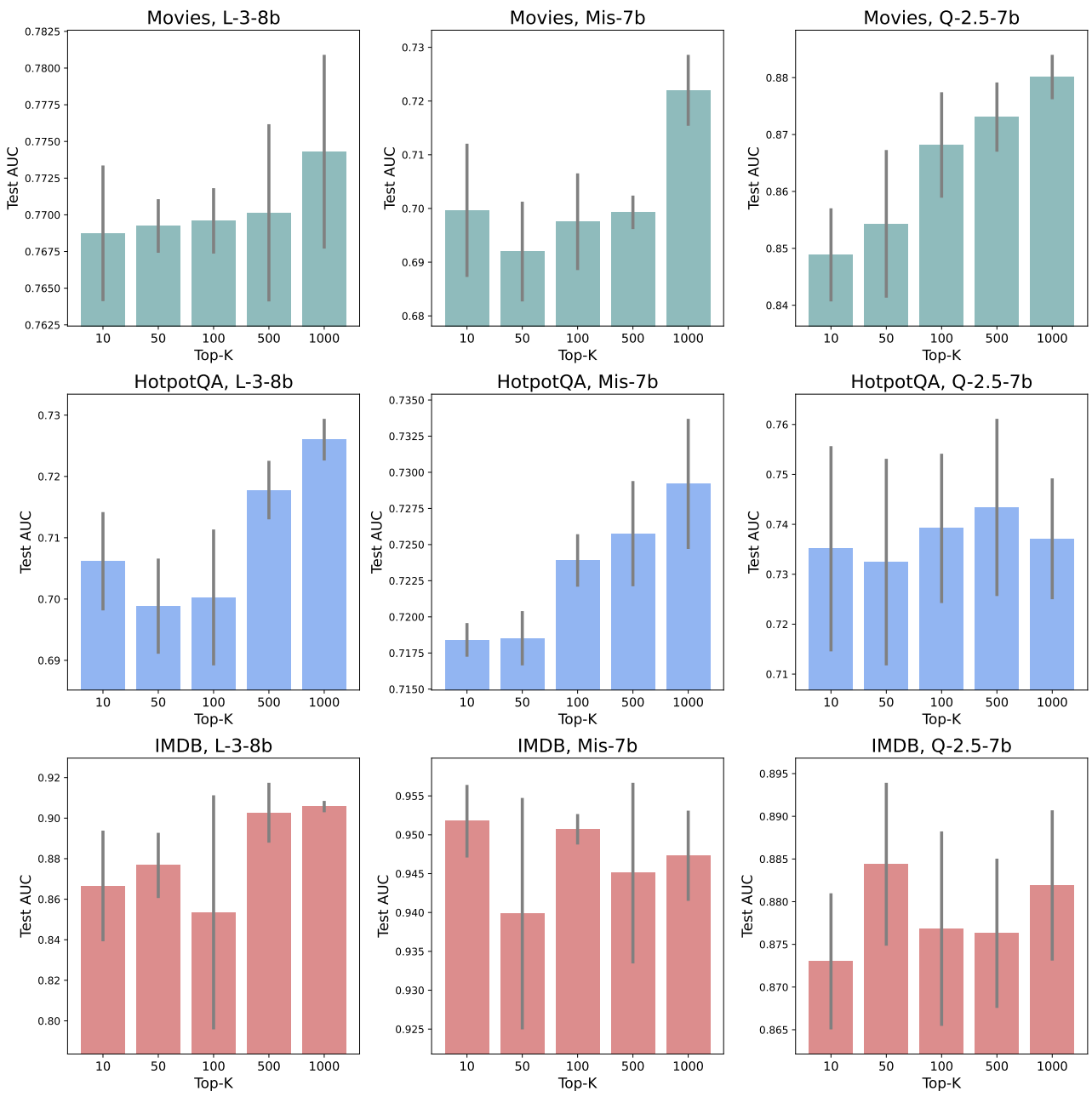

Figure 8: Ablation study analyzing the effect of the hyperparameter K introduced in Equation (1).

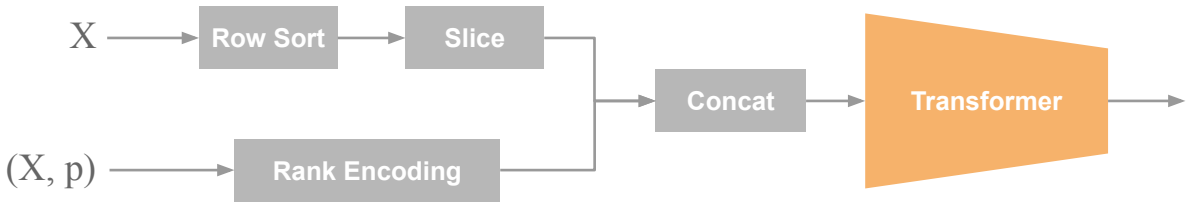

Figure 9: A visualization of LOS-NET .