AP-OOD: Attention Pooling for Out-of-Distribution Detection

Anonymous authors
Paper under double-blind review

Abstract

Out-of-distribution (OOD) detection, which maps high-dimensional data into a scalar OOD score, is critical for the reliable deployment of machine learning models. A key challenge in recent research is how to effectively leverage and aggregate token embeddings from language models to obtain the OOD score. In this work, we propose AP-OOD, a novel OOD detection method for natural language that goes beyond simple average-based aggregation by exploiting token-level information. AP-OOD is a semi-supervised approach that flexibly interpolates between unsupervised and supervised settings, enabling the use of limited auxiliary outlier data. Empirically, AP-OOD sets a new state of the art in OOD detection for text: in the unsupervised setting, it reduces the FPR95 (false positive rate at 95% true positives) from 27.77% to 5.91% on XSUM summarization, and from 75.19% to 68.13% on WMT15 En–Fr translation.

1 Introduction

Out-of-distribution (OOD) detection is essential for deploying machine learning models in the real world. In practical settings many models encounter inputs that deviate from the model's training distribution. For example, a model trained to summarize news articles might also receive a prompt with a cooking recipe. In such situations, models may assign unwarranted confidence to their predictions, leading to erroneous outputs and hallucination. A hallucination is a state in which the model generates output that is nonsensical or unfaithful to the prompt (Farquhar et al., 2024). For example, Ren et al. (2023) observe that a common failure case in abstractive summarization is for the model to output "All images are copyrighted" when prompted to summarize news articles from a publisher (CNN) that differs from what it was trained on (BBC). Many authors attribute hallucination to model uncertainty (e.g., Farquhar et al., 2024; Aichberger et al., 2025), which decomposes into aleatoric uncertainty (resulting from noise in the data) and epistemic uncertainty (resulting from a lack of training data). OOD prompts exhibit high epistemic uncertainty (Ren et al., 2023). The purpose of OOD detection is to classify these inputs as OOD such that the system can then, for instance, notify the user that no output can be generated. Many existing post-hoc OOD detection methods (e.g., Huang et al., 2021; Sun & Li, 2022; Wang et al., 2022) assume a classifier as the base model. In contrast, in language modeling, the base model is typically an autoregressive generative model without an explicit classification head. This necessitates the development of OOD detection methods specifically tailored for language modeling, and we believe that the OOD detection community can benefit from generative language modeling as an additional benchmark. Our contributions are as follows:

1. We propose AP-OOD, an OOD detection approach for natural language that leverages token-level information to detect OOD sequences.
2. AP-OOD is a semi-supervised approach: It can be applied in unsupervised (i.e., when there exists no knowledge about OOD samples) and supervised settings (i.e., when some OOD data of interest is available to the practitioner), and smoothly interpolates between the two.
3. We show that AP-OOD can improve OOD detection for natural language in summarization and translation.

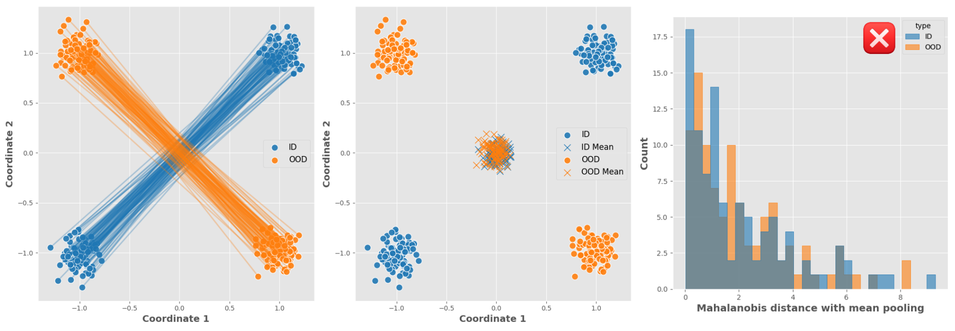


Figure 1: Illustrative example for the failure of mean pooling. **(Left)** ID and OOD sequences $\boldsymbol{Z}_i \in \mathbb{R}^{2\times 2}$, where each sequence contains a pair of token embeddings with two features each. Token embeddings that belong to the same sequence are connected with lines. **(Center)** The means of the ID and OOD sequences both cluster around the origin. **(Right)** A mean pooling approach cannot discriminate between the ID and OOD sequences.

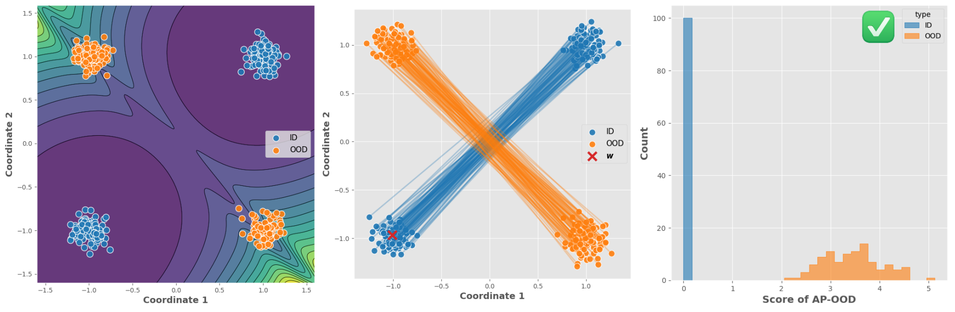


Figure 2: Illustrative example for the mechanism that AP-OOD uses to correctly discriminate between ID and OOD (as opposed to the mean pooling approaches). The setting is the same as in Figure 1. **(Left)** The loss landscape forms two basins at the locations of the ID token embeddings. **(Center)** After training AP-OOD with a single weight vector $\boldsymbol{w}$, the learned $\boldsymbol{w}$ is located in one of the basins. **(Right)** AP-OOD achieves perfect discrimination between the ID and OOD sequences.

4. We provide a theoretical motivation for the suitability of AP-OOD for OOD detection on tokenized data.

1.1 Background

Consider a language model trained that given input sequences $(\boldsymbol{x}_1, \ \boldsymbol{x}_2, \ \dots, \ \boldsymbol{x}_N)$ with $\boldsymbol{x}_i \in \mathcal{X}$[1] autoregressively generates target sequences $(\boldsymbol{y}_1, \ \boldsymbol{y}_2, \ \dots, \ \boldsymbol{y}_N)$ with $\boldsymbol{y}_i \in \mathcal{X}$. The input sequences are drawn i.i.d.: $\boldsymbol{x}_i \sim p_{\text{ID}}$. We consider input sequences $\boldsymbol{x} \in \mathcal{X}$ that deviate considerably from the data generation $p_{\text{ID}}(\boldsymbol{x})$ that defines the "normality" of our data as OOD. Following Ruff et al. (2021), an observed sequence is OOD if it is an element of the set

$$\mathbb{O} \ := \ \{\boldsymbol{x} \in \mathcal{X} \mid p_{\text{ID}}(\boldsymbol{x}) < \epsilon\} \text{ where } \epsilon \geq 0, \tag{1}$$

and $\epsilon \in \mathbb{R}$ is a density threshold. In practice, it is common (e.g., Hendrycks & Gimpel, 2016; Lee et al., 2018; Hofmann et al., 2024) to define a score $s : \mathcal{Z} \to \mathbb{R}$ that uses an encoder $\phi : \mathcal{X} \to \mathcal{Z}$ (where $\mathcal{Z}$ denotes an embedding space). Given s and ϕ, OOD detection can be formulated as a binary classification task with the classes in-distribution (ID) and OOD:

$$\hat{B}(\boldsymbol{x}, \gamma) \ = \ \begin{cases} \text{ID} & \text{if } s(\phi(\boldsymbol{x})) \geq \gamma \\ \text{OOD} & \text{if } s(\phi(\boldsymbol{x})) < \gamma \end{cases}. \tag{2}$$

The outlier score should — in the best case — preserve the density ranking, but it does not have to fulfill all requirements of a probability density (proper normalization or nonnegativity).

[1]We use $\mathcal{X} := \bigcup_{S\geq 1} \mathcal{V}^S$ for the set of input sequences, and $\mathcal{V} := \{v_1, \dots, v_V\}$ is the vocabulary.

For evaluation, the threshold $\gamma \in \mathbb{R}$ is typically chosen such that 95% of ID samples from a previously unseen validation set are correctly classified as ID. However, metrics like the area under the receiver operating characteristic (AUROC) can be directly computed on $s(\phi(\boldsymbol{x}))$ without fixing γ, since the AUROC sweeps over all possible thresholds.

2 Method

AP-OOD is a semi-supervised method: It can be trained without access to outlier data (unsupervised), and with access to outlier data (supervised), and can smoothly transition between those two scenarios as more outlier data becomes available for training. In the following, we first introduce AP-OOD in an unsupervised scenario (Section 2.1) and generalize it to the supervised scenario (Section 2.2).

2.1 Unsupervised OOD Detection

Background Ren et al. (2023) propose to detect OOD inputs using token embeddings obtained from a transformer encoder–decoder model (Vaswani et al., 2017b) trained on the language modeling task. Given an input sequence $\boldsymbol{x}$, they obtain a sequence of token embeddings. They compare obtaining embeddings $\boldsymbol{E} \in \mathcal{Z}$[2] from the encoder $\phi_{\text{enc}} : \mathcal{X} \to \mathcal{Z}$ and generating a sequence of embeddings $\boldsymbol{G} \in \mathcal{Z}$ using the decoder $\phi_{\text{dec}} : \mathcal{Z} \to \mathcal{Z}$:

$$\boldsymbol{E} \ := \ \phi_{\text{enc}}(\boldsymbol{x}) \qquad \boldsymbol{G} \ := \ \phi_{\text{dec}}(\boldsymbol{E}). \tag{3}$$

For clarity, we write $\boldsymbol{Z} \in \mathcal{Z}$ for a sequence of token embeddings, whether produced by the encoder or the decoder, and we call $\boldsymbol{Z}$ the sequence representation of $\boldsymbol{x}$. To obtain a single vector $\bar{\boldsymbol{z}} \in \mathbb{R}^D$, Ren et al. (2023) perform mean pooling:

$$\bar{\boldsymbol{z}} \ := \ \frac{1}{S} \sum_{s=1}^{S} \boldsymbol{z}_s. \tag{4}$$

Then, they propose to measure whether $\bar{\boldsymbol{z}}$ is OOD by first fitting a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\boldsymbol{\mu} \in \mathbb{R}^D$, $\boldsymbol{\Sigma} \in \mathbb{R}^{D \times D}$ to the per-sequence mean embeddings computed from the training corpus, and then computing the squared Mahalanobis distance between $\bar{\boldsymbol{z}}$ and $\boldsymbol{\mu}$:

$$d^2_{\text{Maha}}(\bar{\boldsymbol{z}}, \boldsymbol{\mu}) \ := \ (\bar{\boldsymbol{z}} \ - \ \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\bar{\boldsymbol{z}} \ - \ \boldsymbol{\mu}) \quad \text{and} \quad s_{\text{Maha}}(\bar{\boldsymbol{z}}) \ := \ - \ d^2_{\text{Maha}}(\bar{\boldsymbol{z}}, \boldsymbol{\mu}). \tag{5}$$

Averaging hides anomalies. The key limitation of the approach described above is the use of the **mean** of the token embeddings $\boldsymbol{Z}$: Averaging the entire sequence into the mean $\bar{\boldsymbol{z}}$ discards the token-level structure that would otherwise be informative for detecting whether a sequence is OOD. Figure 1 shows a toy example of this failure mode: The ID and OOD sequences are indistinguishable using their means, and therefore, the Mahalanobis distance with mean pooling fails to discriminate between them.

Mahalanobis decomposition. To address this limitation, we begin by expressing the Mahalanobis distance as a directional decomposition:

$$d^2_{\text{Maha}}(\bar{\boldsymbol{z}}, \boldsymbol{\mu}) \ = \ \sum_{j=1}^{D} \left(\boldsymbol{w}_j^T \bar{\boldsymbol{z}} - \ \boldsymbol{w}_j^T \boldsymbol{\mu} \right)^2, \tag{6}$$

The weight vectors $\boldsymbol{w}_j \in \mathbb{R}^D$ form a basis of $\mathbb{R}^D$ and determine $\boldsymbol{\Sigma}^{-1}$ via $\boldsymbol{\Sigma}^{-1} = \sum_{j=1}^{D} \boldsymbol{w}_j \boldsymbol{w}_j^T$. One possibility to map a given $\boldsymbol{\Sigma}^{-1}$ to weight vectors $\boldsymbol{w}_j$ is to select the directions of the $\boldsymbol{w}_j$ as the unit-norm eigenvectors of $\boldsymbol{\Sigma}^{-1}$, and to select the squared norms of the $\boldsymbol{w}_j$ as their corresponding eigenvalues (see Appendix B.2).

[2]We use $\mathcal{Z} := \bigcup_{S \geq 1} \mathbb{R}^{D \times S}$ for all finite-length sequences of D-dimensional token embeddings.

Algorithm 1 AP-OOD

```
Require: $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$, $\phi_{\text{enc}}$, $\phi_{\text{dec}}$, $\beta$, $M$, nsteps
 1: for $i = 1$ to $N$ do
 2:     Compute sequence embedding $\boldsymbol{Z}_i$ using $\boldsymbol{Z}_i \leftarrow \phi_{\text{enc}}(\boldsymbol{x}_i)$ or $\boldsymbol{Z}_i \leftarrow \phi_{\text{dec}}(\phi_{\text{enc}}(\boldsymbol{x}_i))$.
 3: for step $= 1$ to nsteps do
 4:     Randomly sample mini-batch indices $\mathcal{B} \subset \{1, \ldots, N\}$
 5:     Collect mini-batch $\{\boldsymbol{Z}_i\}_{i \in \mathcal{B}}$.
 6:     Form batch-local concatenation $\tilde{\boldsymbol{Z}}_B \leftarrow \|_{i \in \mathcal{B}} \boldsymbol{Z}_i$.
 7:     Compute loss $\mathcal{L} \leftarrow \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} d^2(\boldsymbol{Z}_i, \tilde{\boldsymbol{Z}}_B) - \sum_{j=1}^{M} \log(||\boldsymbol{w}_j||_2^2)$.
 8:     Compute gradients of $\mathcal{L}$ w.r.t. $(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_M)$ and perform a gradient update
 9: Do mini-batch attention pooling to compute $\boldsymbol{\mu}_j \leftarrow \tilde{\boldsymbol{Z}}\mathrm{softmax}(\beta\ \tilde{\boldsymbol{Z}}^T \boldsymbol{w}_j)$ (Appendix C.1)
10: $s(\boldsymbol{Z}) \leftarrow \sum_{j=1}^{M} -d_j^2(\boldsymbol{Z}, \tilde{\boldsymbol{Z}}) + \log\left(||\boldsymbol{w}_j||_2^2\right)$.
11: return $s(\cdot)$
```

Beyond mean pooling. To overcome the limitations of mean pooling, we generalize Equation (6) by using attention pooling (Bahdanau, 2014; Ramsauer et al., 2021):

$$\mathrm{AttPool}_\beta(\boldsymbol{Z}, \boldsymbol{w}) \ := \ \boldsymbol{Z}\mathrm{softmax}(\beta\ \boldsymbol{Z}^T \boldsymbol{w}) \quad \text{and} \quad \bar{\boldsymbol{z}} \ := \ \mathrm{AttPool}_\beta(\boldsymbol{Z}, \boldsymbol{w}). \tag{7}$$

where $\beta \in \mathbb{R}_{\geq 0}$ is the inverse temperature, and $\boldsymbol{w} \in \mathbb{R}^D$ is a learnable query. AP-OOD also uses attention for the corpus-wide pooling: Given the sequence representations $(\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_N)$ where $\boldsymbol{Z}_i \in \mathcal{Z}$ from a corpus $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$ with $\boldsymbol{Z}_i := \phi_{\text{enc}}(\boldsymbol{x}_i)$, we define $\tilde{\boldsymbol{Z}} \in \mathcal{Z}$ as the concatenation of all sequence representations: $\tilde{\boldsymbol{Z}} := (\boldsymbol{Z}_1 \parallel \cdots \parallel \boldsymbol{Z}_N)$. AP-OOD estimates $\boldsymbol{\mu} := \mathrm{AttPool}_\beta(\tilde{\boldsymbol{Z}}, \boldsymbol{w})$. Given the $\bar{\boldsymbol{z}}$ and $\boldsymbol{\mu}$ from the attention pooling, AP-OOD estimates $d^2(\boldsymbol{Z}, \tilde{\boldsymbol{Z}})$, the squared distance between a sequence representation $\boldsymbol{Z}$ and the concatenation $\tilde{\boldsymbol{Z}}$ analogous to Equation (6):

$$d^2(\boldsymbol{Z}, \tilde{\boldsymbol{Z}}) := \sum_{j=1}^{M} \Big(\boldsymbol{w}_j^T \boldsymbol{Z}\mathrm{softmax}(\beta\ \boldsymbol{Z}^T \boldsymbol{w}_j) - \boldsymbol{w}_j^T \tilde{\boldsymbol{Z}}\mathrm{softmax}(\beta\ \tilde{\boldsymbol{Z}}^T \boldsymbol{w}_j) \Big)^2 = \sum_{j=1}^{M} d_j^2(\boldsymbol{Z}, \tilde{\boldsymbol{Z}}). \tag{8}$$

We refer to $M \in \mathbb{N}$ as the number of heads. In general, M does not need to equal the embedding dimension D. We show in Appendix B.3 that, when $\beta = 0$ and $M = D$, Equation (8) reduces to the Mahalanobis distance (Equations (5) and (6)). To the best of our knowledge, AP-OOD is the first approach to integrate attention pooling into the Mahalanobis distance via a learnable directional decomposition. In Appendix *B*.1, we show that $s_{\min}(\boldsymbol{Z}) = \min_j -d_j^2(\boldsymbol{Z}, \tilde{\boldsymbol{Z}}) + \log(||\boldsymbol{w}_j||_2^2)$ is a score function as defined in Equation (2). Our score arises naturally as the upper bound

$$s(\boldsymbol{Z}) \ := \ \sum_{j=1}^{M} -d_j^2(\boldsymbol{Z}, \tilde{\boldsymbol{Z}}) \ + \ \log(||\boldsymbol{w}_j||_2^2). \tag{9}$$

In Appendix D.7, we empirically compare the min-based score $s_{\min}(\boldsymbol{Z})$ to its upper-bound variant $s(\boldsymbol{Z})$ and find that $s(\boldsymbol{Z})$ yields stronger OOD discrimination. The choice of this score naturally leads to the loss function of AP-OOD:

$$\mathcal{L}(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_M) \ := \ \frac{1}{N} \sum_{i=1}^{N} d^2(\boldsymbol{Z}_i, \tilde{\boldsymbol{Z}}) \ - \ \sum_{j=1}^{M} \log\left(||\boldsymbol{w}_j||_2^2\right). \tag{10}$$

We provide the pseudocode for AP-OOD in Algorithm 1. Scaling to large data sets requires efficient computation of $\boldsymbol{\mu} = \tilde{\boldsymbol{Z}}\mathrm{softmax}(\beta\ \tilde{\boldsymbol{Z}}^T \boldsymbol{w})$; the naive method loads the entire concatenated sequence $\tilde{\boldsymbol{Z}}$ into memory, but we reduce the memory footprint by performing attention pooling on mini-batches. We describe this procedure in Appendix C.1.

Multiple queries per head. We now extend AP-OOD and use multiple queries per head. We use a set of stacked queries $\boldsymbol{W}_j = (\boldsymbol{w}_{j1}, \ldots, \boldsymbol{w}_{jT}) \in \mathbb{R}^{D \times T}$ per head. For simplicity, we consider a single head with the queries $\boldsymbol{W} \in \mathbb{R}^{D\times T}$ for now. We begin by extending the softmax notation from Ramsauer et al. (2021) to matrix-valued arguments. Given a matrix $\boldsymbol{A} \in \mathbb{R}^{S\times T}$

$$\text{softmax}(\beta \boldsymbol{A})_{st} := \frac{\exp(\beta a_{st})}{\sum_{s'=1}^{S}\sum_{t'=1}^{T}\exp(\beta a_{s't'})}. \quad (11)$$

In other words, the softmax normalizes over the rows and columns of $\boldsymbol{A}$. Next, we extend the attention pooling process from Equation (7) with the matrix-valued softmax: AP-OOD transforms the sequence representation $\boldsymbol{Z} \in \mathbb{R}^{D\times S}$ with S tokens to a new sequence representation $\bar{\boldsymbol{Z}} \in \mathbb{R}^{D\times T}$ with T tokens using $\bar{\boldsymbol{Z}} := \boldsymbol{Z}\boldsymbol{P}$. The updated attention pooling process is

$$\text{AttPool}_\beta(\boldsymbol{Z}, \boldsymbol{W}) \ := \ \boldsymbol{Z}\text{softmax}(\beta \ \boldsymbol{Z}^T\boldsymbol{W}) \quad \text{and} \quad \bar{\boldsymbol{Z}} \ := \ \text{AttPool}_\beta(\boldsymbol{Z}, \boldsymbol{W}). \quad (12)$$

To the best of our knowledge, this work is the first to use a matrix-valued global softmax to transform a sequence $\boldsymbol{Z}$ into another sequence $\bar{\boldsymbol{Z}}$. Finally, AP-OOD uses $\boldsymbol{W} \in \mathbb{R}^{D\times T}$ to transform the $\bar{\boldsymbol{Z}} \in \mathbb{R}^{D\times T}$ to a real number with the Frobenius inner product $\langle \boldsymbol{W}, \bar{\boldsymbol{Z}}\rangle_\text{F} = \text{vec}(\boldsymbol{W})^T \text{vec}(\bar{\boldsymbol{Z}}) = \text{Tr}(\boldsymbol{W}^T\bar{\boldsymbol{Z}})$. To summarize, the extended squared distance is

$$d^2(\boldsymbol{Z}, \tilde{\boldsymbol{Z}}) \ := \ \sum_{j=1}^{M}\Big(\text{Tr}(\boldsymbol{W}_j^T\boldsymbol{Z}\text{softmax}(\beta \ \boldsymbol{Z}^T\boldsymbol{W}_j)) \ - \ \text{Tr}(\boldsymbol{W}_j^T\tilde{\boldsymbol{Z}}\text{softmax}(\beta \ \tilde{\boldsymbol{Z}}^T\boldsymbol{W}_j))\Big)^2. \quad (13)$$

Finally, the regularizing term is $-\log(||\boldsymbol{W}||_\text{F}^2)$ (where $||\cdot||_\text{F}^2$ denotes the squared Frobenius norm). To summarize, the extended loss is

$$\mathcal{L}(\boldsymbol{W}_1, \dots, \boldsymbol{W}_M) \ := \ \frac{1}{N}\sum_{i=1}^{N} d^2(\boldsymbol{Z}_i, \tilde{\boldsymbol{Z}}) \ - \ \sum_{j=1}^{M}\log\left(||\boldsymbol{W}_j||_\text{F}^2\right). \quad (14)$$

We provide PyTorch-style pseudocode implementing Equation (14) in Appendix C.2.

2.2 Supervised OOD Detection

Background. Supplying an OOD detector with information about the distribution of the OOD examples at training time can improve the ID–OOD decision boundary (Hendrycks et al., 2018). In practice, it is hard to find OOD data for training that is fully indicative of the OOD distribution seen during inference. Outlier exposure (OE; Hendrycks et al., 2018) therefore uses a large and diverse auxiliary outlier set (AUX; e.g., C4 for text data) as a stand-in for the OOD case. However, acquiring such large and diverse AUX datasets is not always possible. For example, consider a translation task with a less widely spoken source language. As another example, consider detecting defects in industrial machines using recordings of their sounds (Nishida et al., 2024). Practitioners can collect a relatively large amount of ID audio data from machines while they run without defects. However, it is much harder to collect diverse AUX examples from defective machines because defects are infrequent. In such a case, one might have to resort to a smaller AUX data set. Therefore, an OOD detector should scale gracefully with the degree of auxiliary supervision, adapting to the available number of AUX examples (e.g., Ruff et al., 2019; Liznerski et al., 2022; Yoon et al., 2023; Ivanov et al., 2024; Qiao et al., 2024).

Utilizing AUX data. To adapt AP-OOD to the supervised setting, we follow Ruff et al. (2019) and Liznerski et al. (2022): AP-OOD punishes large squared distances $d^2(\boldsymbol{Z}, \tilde{\boldsymbol{Z}})$ for ID samples $\boldsymbol{Z}$ and encourages large squared distances for AUX samples $\boldsymbol{Z}$. Formally, AP-OOD minimizes the binary cross-entropy loss with the classes ID and AUX with $p(y = \text{ID}|\boldsymbol{Z}) = \exp(-d^2(\boldsymbol{Z}, \tilde{\boldsymbol{Z}}))$. Given N ID examples $(\boldsymbol{Z}_1, \dots, \boldsymbol{Z}_N)$, and N' AUX examples $(\boldsymbol{Z}_{N+1}, \dots, \boldsymbol{Z}_{N+N'})$, AP-OOD minimizes the supervised loss

$$\mathcal{L}_\text{SUP} \ := \ \frac{1}{N+N'}\sum_{i=1}^{N} d^2(\boldsymbol{Z}_i, \tilde{\boldsymbol{Z}}) \ - \ \lambda \ \frac{1}{N+N'}\sum_{i=N+1}^{N+N'}\log(1 - \exp(-d^2(\boldsymbol{Z}_i, \tilde{\boldsymbol{Z}}))), \quad (15)$$

where $\lambda \in \mathbb{R}_{\geq 0}$. If $\lambda = 0$, $\mathcal{L}_\text{SUP}$ equals the unsupervised loss $\mathcal{L}$ without the regularizing term.

Table 1: Unsupervised OOD detection performance on text summarization. We compare results from AP-OOD, Mahalanobis (Lee et al., 2018; Ren et al., 2023), KNN (Sun et al., 2022), Deep SVDD (Ruff et al., 2018), model perplexity (Ren et al., 2023), and entropy (Malinin & Gales, 2020) on PEGASUS$_{\text{LARGE}}$ trained on XSUM as the ID data set. ↓ indicates "lower is better" and ↑ "higher is better". All values in %. We estimate standard deviations across five independent data set splits and training runs.

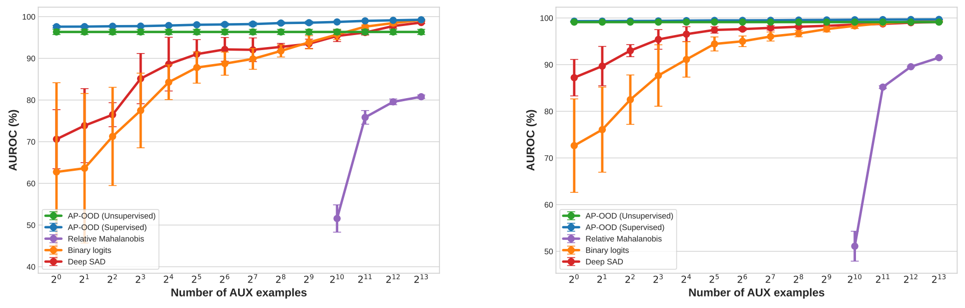
		CNN/DM	Newsroom	Reddit	Samsum	Mean
			Input OOD			
Mahalanobis	AUROC ↑	$69.00^{\pm 0.27}$	$86.37^{\pm 0.19}$	$98.64^{\pm 0.07}$	$\underline{99.77^{\pm 0.01}}$	88.45
	FPR95 ↓	$92.19^{\pm 0.08}$	$64.48^{\pm 0.71}$	$2.45^{\pm 0.34}$	$\underline{0.17^{\pm 0.02}}$	39.82
KNN	AUROC ↑	$54.34^{\pm 0.15}$	$73.76^{\pm 0.09}$	$94.52^{\pm 0.03}$	$98.82^{\pm 0.01}$	80.36
	FPR95 ↓	$99.40^{\pm 0.03}$	$88.56^{\pm 0.17}$	$51.24^{\pm 0.70}$	$3.07^{\pm 0.16}$	60.57
Deep SVDD	AUROC ↑	$\underline{75.86^{\pm 1.00}}$	$\underline{91.20^{\pm 0.21}}$	$\underline{99.73^{\pm 0.05}}$	$99.57^{\pm 0.04}$	$\underline{91.59}$
	FPR95 ↓	$\underline{73.70^{\pm 2.35}}$	$\underline{36.46^{\pm 1.12}}$	$\underline{0.26^{\pm 0.09}}$	$0.67^{\pm 0.17}$	$\underline{27.77}$
AP-OOD (Ours)	AUROC ↑	$\mathbf{96.13^{\pm 0.44}}$	$\mathbf{99.10^{\pm 0.08}}$	$\mathbf{99.91^{\pm 0.03}}$	$\mathbf{99.80^{\pm 0.04}}$	**98.74**
	FPR95 ↓	$\mathbf{19.51^{\pm 2.24}}$	$\mathbf{4.11^{\pm 0.28}}$	$\mathbf{0.00^{\pm 0.01}}$	$\mathbf{0.04^{\pm 0.03}}$	**5.91**
			Output OOD			
Perplexity	AUROC ↑	$42.20^{\pm 0.14}$	$53.99^{\pm 0.31}$	$83.38^{\pm 0.15}$	$78.53^{\pm 0.31}$	64.52
	FPR95 ↓	$77.71^{\pm 0.17}$	$79.07^{\pm 0.57}$	$45.56^{\pm 0.40}$	$46.96^{\pm 0.20}$	62.32
Entropy	AUROC ↑	$59.59^{\pm 0.21}$	$77.20^{\pm 0.52}$	$93.47^{\pm 0.21}$	$87.17^{\pm 0.20}$	79.36
	FPR95 ↓	$79.04^{\pm 0.75}$	$64.24^{\pm 1.21}$	$30.19^{\pm 1.34}$	$50.47^{\pm 1.64}$	55.98
Mahalanobis	AUROC ↑	$63.27^{\pm 0.17}$	$88.26^{\pm 0.11}$	$97.40^{\pm 0.09}$	$97.29^{\pm 0.08}$	86.55
	FPR95 ↓	$89.84^{\pm 0.13}$	$47.83^{\pm 0.71}$	$11.13^{\pm 0.58}$	$13.57^{\pm 0.25}$	40.59
KNN	AUROC ↑	$\underline{74.37^{\pm 0.13}}$	$86.96^{\pm 0.08}$	$95.85^{\pm 0.06}$	$\underline{97.33^{\pm 0.03}}$	88.63
	FPR95 ↓	$\underline{73.36^{\pm 0.20}}$	$53.44^{\pm 0.58}$	$15.78^{\pm 0.27}$	$\underline{10.29^{\pm 0.22}}$	38.22
Deep SVDD	AUROC ↑	$68.31^{\pm 1.63}$	$\mathbf{94.13^{\pm 0.12}}$	$\underline{97.60^{\pm 0.26}}$	$95.97^{\pm 0.15}$	$\underline{89.00}$
	FPR95 ↓	$76.76^{\pm 1.15}$	$\mathbf{19.22^{\pm 0.34}}$	$\underline{8.90^{\pm 1.25}}$	$20.17^{\pm 1.28}$	$\underline{31.26}$
AP-OOD (Ours)	AUROC ↑	$\mathbf{93.37^{\pm 0.54}}$	$\underline{92.62^{\pm 0.67}}$	$\mathbf{98.04^{\pm 0.28}}$	$\mathbf{98.30^{\pm 0.11}}$	**95.59**
	FPR95 ↓	$\mathbf{23.12^{\pm 1.97}}$	$\underline{29.91^{\pm 2.93}}$	$\mathbf{6.34^{\pm 1.56}}$	$\mathbf{6.83^{\pm 0.64}}$	**16.55**

3 Experiments

Toy experiment. We present a toy experiment illustrating the main intuitions behind AP-OOD. Figure 1 demonstrates a simple failure mode of mean pooling approaches: First, we generate ID and OOD token embeddings $\boldsymbol{Z}_i \in \mathbb{R}^{2\times 2}$. Each ID sequence representation consists of one token sampled from $\mathcal{N}((1,1),\ \sigma^2\boldsymbol{I})$ (where $\sigma := 0.1$) and one token sampled from $\mathcal{N}((-1,-1),\ \sigma^2\boldsymbol{I})$. The OOD sequences contain two tokens sampled from $\mathcal{N}((-1,1),\ \sigma^2\boldsymbol{I})$ and $\mathcal{N}((1,-1),\ \sigma^2\boldsymbol{I})$, respectively. The left panel shows the generated sequences, where each sequence consists of two dots (representing the two tokens) connected by a line. Because the means of the ID and OOD sequences both cluster around the origin (central panel), the Mahalanobis distance with mean pooling fails to discriminate between them (right panel). Figure 2 shows how AP-OOD overcomes this limitation: We set $M = 1$ and $T = 1$ and train AP-OOD as described in Section 2.1 on the ID data only, but we modify the pooling mechanism from Equation (7): We replace the dot product similarity in the softmax with the negative squared Euclidean distance, as it is known to work better in low-dimensional spaces (we provide the formal definition for this modification in Appendix D.1). The left panel of Figure 2 shows that the loss landscape of $\boldsymbol{w}$ forms two basins at the locations of the ID tokens. The central panel shows that after training, $\boldsymbol{w}$ is located in one of the basins. Finally, the right panel shows that AP-OOD perfectly discriminates ID and OOD.

Summarization. We follow Ren et al. (2023) and use a PEGASUS$_{\text{LARGE}}$ (Zhang et al., 2020) fine-tuned on the ID data set XSUM (Narayan et al., 2018). We utilize the C4 training split as the AUX data set. We measure the OOD detection performance on the data sets CNN/Daily Mail (CNN/DM; news articles from CNN and Daily Mail; Hermann et al., 2015; See et al., 2017), Newsroom (articles and summaries written by authors and editors from 38 news publications; Grusky et al., 2018), Reddit TIFU (posts and summaries from the online discussion forum Reddit; Kim et al., 2018), and Samsum (summaries of casual dialogues; Gliwa et al., 2019). The ForumSum data set used in the experiments of Ren et al. (2023) has been retracted. Therefore, we do not use it in our experiments.

Translation. We train a Transformer (base) on WMT15 En–Fr (Bojar et al., 2015). The model trains for 100,000 steps using AdamW (Loshchilov & Hutter, 2017) with a cosine schedule (Loshchilov & Hutter, 2016), linear warmup, and a peak learning rate of 5×10^{-4}.



(a) CNN/DM (b) Newsroom

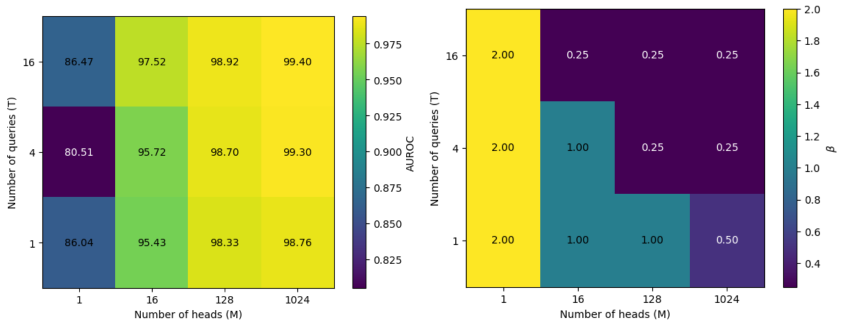
Figure 3: OOD detection performance on the input token embeddings of PEGASUS$_{\text{LARGE}}$ trained on XSUM. We vary the number of AUX samples and compare AP-OOD, binary logits (Ren et al., 2023), Deep SAD (Ruff et al., 2019), and relative Mahalanobis (Ren et al., 2023). AP-OOD attains the highest AUROC independent of AUX sample count.

We set the batch size to 1024 and the context length to 512. Following Ren et al. (2023), the AUX data set is ParaCrawl En–Fr, and the OOD data sets are newstest2014 (nt2014), newsdiscussdev2015 (ndd2015), and newsdiscusstest2015 (ndt2015) from WMT15 (Bojar et al., 2015), and the Law, Koran, Medical, IT, and Subtitles subsets from OPUS (Tiedemann, 2012; Aulamo & Tiedemann, 2019).

Training. We extract 100,000 ID sequence representations ($\boldsymbol{E}$ or $\boldsymbol{G}$) and use all extracted representations for training AP-OOD in all experiments. We also extract AUX sequence representations, and we vary the number of AUX sequences available from 0 (unsupervised) to 10,000 (fully supervised). While training AP-OOD, the transformer model remains frozen. We use the Adam optimizer (Kingma & Ba, 2014) without weight decay, set the learning rate to 0.01, and apply a cosine schedule (Loshchilov & Hutter, 2016). We train for 2,000 steps with a batch size of 512. We select M and T such that the parameter count of AP-OOD matches the parameter count of the Mahalanobis method (i.e., the size of $\boldsymbol{\Sigma}$). For more information on hyperparameter selection, we refer to Appendix D.2. An additional scaling experiment on input sequence representations of the summarization task investigates larger parameter spaces: We train on the full XSUM data set (instead of the 100,000 ID sequence representations used in the other experiments). We select the number of heads (M) from the set $\{1, 16, 128, 1024\}$, the number of queries (T) from the set $\{1, 4, 16\}$, and β from $\{1/\sqrt{D}, 0.25, 0.5, 1, 2\}$. The largest configuration has a parameter count 16 times greater than the Mahalanobis baseline.

Baselines. We compare AP-OOD to six unsupervised OOD detection methods: We apply the embedding-based methods Mahalanobis (Lee et al., 2018; Ren et al., 2023), KNN (Sun et al., 2022), and Deep SVDD (Ruff et al., 2018) to both the input and output sequence representations ($\boldsymbol{E}$ and $\boldsymbol{G}$, respectively), and we apply Perplexity (Ren et al., 2023) and Entropy (Malinin & Gales, 2020) to the output of the decoder. We also compare AP-OOD to three supervised OOD detection methods: binary logits (Ren et al., 2023), relative Mahalanobis (Ren et al., 2023), and Deep SVDD (Ruff et al., 2019). We evaluate the discriminative power of the methods in our comparison using the false positive rate at 95% true positives (FPR95) and AUROC.

Audio data. To demonstrate the effectiveness of AP-OOD on data modalities other than text, we apply the method to the MIMII-DG audio data set (Dohi et al., 2022). The data set comprises audio recordings of 15 different machines, ranging from 10 to 12 seconds in length. The dataset contains 990 samples per machine. During preprocessing, the raw audio waveforms are converted into audio spectrograms. We train a transformer to classify a subset of 7 machines. The remaining 8 machines are considered as OOD. The architecture and training method for the network were adopted from Huang et al. (2022). To adjust for the



(a) Mean AUROC (b) Best β

Figure 4: OOD detection performance on the input token embeddings of PEGASUS$_{\text{LARGE}}$ trained on XSUM when scaling to large M and T. We vary $M \in \{1, 16, 128, 1024\}$, $T \in \{1, 4, 16\}$, and $\beta \in \{1/\sqrt{D}, 0.25, 0.5, 1, 2\}$. **(Left)** Mean AUROC in % for the best β at each (M, T) combination. **(Right)** The best β selected at each (M, T) combination.

small data set size, we decrease the size of the architecture: We increase the patch size to 32×32 pixels, decrease the embedding dimension to 32, and utilize only three attention blocks with four heads each. Consequently, the encoder of the network produces 128 tokens with $D = 32$ features. We train AP-OOD on the encoder output in the unsupervised setting using $M = 128$ and $T = 8$.

4 Results

Table 1 shows the results on unsupervised OOD detection on the text summarization task. AP-OOD surpasses methods with mean pooling by a large margin for both input and output settings for most OOD data sets. Most notably, the mean FPR95 on CNN/DM improves from 73.70% for the best baseline Deep SVDD to 19.51% for AP-OOD. The table also shows that the embedding-based methods (Mahalanobis, KNN, Deep SVDD, and AP-OOD) perform better than the prediction-based baselines perplexity and entropy. Figure 3 shows the results of AP-OOD in the semi-supervised setting: supplying AUX data to AP-OOD improves the AUROC, and more AUX data results in a larger improvement. AP-OOD attains the highest AUROC independent of AUX sample count. We include the results on additional OOD data sets in the semi-supervised setting and results on fully supervised OOD detection on the summarization task in Appendix D.3, and we present ablations on AP-OOD on text summarization in Appendix D.8.

Figure 4 shows the results when scaling AP-OOD to larger parameter counts. As we increase the number of heads (M) and queries (T), we observe a steady increase in the mean AUROC on the summarization task. The highest Mean AUROC of 99.40% is achieved by the largest configuration tested ($M = 1024$, $T = 16$).

Table 2 shows the results on unsupervised OOD detection on the translation task. AP-OOD gives the best average results for the input and output settings. It is noteworthy that in the translation task, the prediction-based methods perform better, with the perplexity baseline outperforming all embedding-based methods evaluated on the output token embeddings except AP-OOD. We hypothesize that this discrepancy can be explained as follows: In translation, ID uncertainty is typically low because the source sentence largely dictates what must be generated — specific words, names, and inflections — so ID perplexities are small and tightly clustered. In text summarization, ID uncertainty is higher because many different summaries can be equally valid, with freedom in what to include and how to phrase it. This raises and spreads ID perplexity and weakens ID–OOD separation when using perplexity. We include results on fully supervised OOD detection for translation in Appendix D.5. Furthermore, we verify the effectiveness of AP-OOD on the decoder-only language modeling paradigm using Pythia-160M in Appendix D.6.

Table 2: Unsupervised OOD detection performance on English-to-French translation. We compare results from AP-OOD, Mahalanobis (Lee et al., 2018; Ren et al., 2023), KNN (Sun et al., 2022), Deep SVDD (Ruff et al., 2018), model perplexity (Ren et al., 2023), and entropy (Malinin & Gales, 2020) on a Transformer (base) trained on WMT15 En–Fr as the ID data set. ↓ indicates "lower is better" and ↑ "higher is better". All values in %. We estimate standard deviations across five independent data set splits and training runs.

		IT	Koran	Law	Medical	Subtitles	ndd2015	ndt2015	nt2014	Mean
					Input OOD					
Mahalanobis	AUROC ↑	93.94$^{\pm 0.01}$	66.82$^{\pm 0.29}$	49.39$^{\pm 0.30}$	78.50$^{\pm 0.41}$	**89.61$^{\pm 0.09}$**	65.87$^{\pm 0.01}$	66.44$^{\pm 0.01}$	51.53$^{\pm 0.01}$	70.26
	FPR95 ↓	31.29$^{\pm 0.29}$	93.46$^{\pm 0.27}$	91.26$^{\pm 0.50}$	63.13$^{\pm 0.77}$	**59.60$^{\pm 0.48}$**	87.01$^{\pm 0.14}$	89.09$^{\pm 0.10}$	97.13$^{\pm 0.10}$	76.50
KNN	AUROC ↑	94.16$^{\pm 0.01}$	66.16$^{\pm 0.24}$	46.68$^{\pm 0.22}$	79.62$^{\pm 0.41}$	89.16$^{\pm 0.11}$	64.81$^{\pm 0.05}$	65.63$^{\pm 0.05}$	53.21$^{\pm 0.05}$	69.93
	FPR95 ↓	32.44$^{\pm 0.12}$	94.69$^{\pm 0.28}$	92.71$^{\pm 0.34}$	67.04$^{\pm 0.73}$	63.35$^{\pm 0.32}$	88.91$^{\pm 0.07}$	89.97$^{\pm 0.04}$	97.51$^{\pm 0.03}$	78.33
Deep SVDD	AUROC ↑	92.53$^{\pm 0.15}$	64.12$^{\pm 0.81}$	**51.56$^{\pm 1.21}$**	77.40$^{\pm 0.52}$	87.64$^{\pm 0.37}$	63.30$^{\pm 0.40}$	63.58$^{\pm 0.31}$	49.31$^{\pm 0.31}$	68.68
	FPR95 ↓	39.37$^{\pm 0.94}$	95.24$^{\pm 0.28}$	92.80$^{\pm 0.29}$	66.17$^{\pm 0.71}$	65.53$^{\pm 1.33}$	89.87$^{\pm 0.22}$	90.91$^{\pm 0.27}$	98.07$^{\pm 0.19}$	79.74
AP-OOD (Ours)	AUROC ↑	**94.88$^{\pm 0.08}$**	**73.51$^{\pm 0.33}$**	51.11$^{\pm 0.38}$	**81.80$^{\pm 0.35}$**	89.14$^{\pm 0.32}$	**69.98$^{\pm 0.15}$**	**70.40$^{\pm 0.27}$**	**57.82$^{\pm 0.23}$**	**73.58**
	FPR95 ↓	**25.00$^{\pm 0.59}$**	**87.48$^{\pm 0.33}$**	**89.45$^{\pm 0.67}$**	**58.51$^{\pm 0.60}$**	60.78$^{\pm 2.07}$	**86.45$^{\pm 0.91}$**	**87.05$^{\pm 0.32}$**	**94.19$^{\pm 0.41}$**	**73.61**
					Output OOD					
Perplexity	AUROC ↑	94.06$^{\pm 0.00}$	77.05$^{\pm 0.20}$	45.18$^{\pm 0.38}$	75.41$^{\pm 0.42}$	92.38$^{\pm 0.08}$	75.32$^{\pm 0.02}$	75.81$^{\pm 0.02}$	61.74$^{\pm 0.02}$	74.62
	FPR95 ↓	35.36$^{\pm 0.01}$	90.54$^{\pm 0.35}$	90.14$^{\pm 0.34}$	69.17$^{\pm 0.60}$	50.11$^{\pm 0.58}$	83.94$^{\pm 0.04}$	85.47$^{\pm 0.00}$	96.80$^{\pm 0.00}$	75.19
Entropy	AUROC ↑	71.44$^{\pm 0.22}$	**86.14$^{\pm 0.32}$**	53.98$^{\pm 0.23}$	51.12$^{\pm 0.44}$	70.95$^{\pm 0.47}$	75.11$^{\pm 0.96}$	72.96$^{\pm 0.22}$	**71.31$^{\pm 0.17}$**	69.13
	FPR95 ↓	71.19$^{\pm 0.95}$	**56.19$^{\pm 1.91}$**	93.94$^{\pm 0.37}$	90.27$^{\pm 0.64}$	74.56$^{\pm 1.23}$	**76.28$^{\pm 2.13}$**	**77.65$^{\pm 1.54}$**	**85.71$^{\pm 1.32}$**	78.23
Mahalanobis	AUROC ↑	90.74$^{\pm 0.01}$	69.38$^{\pm 0.17}$	52.25$^{\pm 0.14}$	75.68$^{\pm 0.47}$	86.57$^{\pm 0.08}$	62.28$^{\pm 0.03}$	62.76$^{\pm 0.02}$	48.63$^{\pm 0.02}$	68.54
	FPR95 ↓	57.02$^{\pm 0.44}$	94.26$^{\pm 0.23}$	97.15$^{\pm 0.15}$	81.34$^{\pm 0.33}$	76.16$^{\pm 0.79}$	93.09$^{\pm 0.29}$	93.93$^{\pm 0.13}$	98.00$^{\pm 0.09}$	86.37
KNN	AUROC ↑	95.35$^{\pm 0.04}$	71.55$^{\pm 0.17}$	**57.40$^{\pm 0.14}$**	78.53$^{\pm 0.58}$	87.06$^{\pm 0.12}$	67.16$^{\pm 0.12}$	67.90$^{\pm 0.13}$	58.38$^{\pm 0.10}$	72.92
	FPR95 ↓	27.61$^{\pm 0.31}$	94.13$^{\pm 0.11}$	93.82$^{\pm 0.32}$	65.10$^{\pm 0.58}$	72.73$^{\pm 0.43}$	91.33$^{\pm 0.08}$	91.88$^{\pm 0.10}$	96.79$^{\pm 0.05}$	79.17
Deep SVDD	AUROC ↑	89.20$^{\pm 0.13}$	67.28$^{\pm 0.80}$	54.40$^{\pm 0.83}$	73.96$^{\pm 0.65}$	84.00$^{\pm 0.19}$	60.37$^{\pm 0.57}$	60.66$^{\pm 0.37}$	47.11$^{\pm 0.22}$	67.12
	FPR95 ↓	62.41$^{\pm 1.21}$	95.19$^{\pm 0.48}$	95.03$^{\pm 0.65}$	81.50$^{\pm 1.69}$	81.56$^{\pm 1.15}$	93.93$^{\pm 0.26}$	95.75$^{\pm 0.44}$	98.41$^{\pm 0.16}$	87.97
AP-OOD (Ours)	AUROC ↑	**96.28$^{\pm 0.11}$**	80.70$^{\pm 0.50}$	53.07$^{\pm 0.68}$	**80.84$^{\pm 0.87}$**	**93.88$^{\pm 0.36}$**	**80.64$^{\pm 0.57}$**	**81.39$^{\pm 0.56}$**	68.12$^{\pm 0.65}$	**79.36**
	FPR95 ↓	**21.20$^{\pm 0.65}$**	82.49$^{\pm 1.29}$	**87.38$^{\pm 0.44}$**	**63.67$^{\pm 1.03}$**	**40.27$^{\pm 3.02}$**	77.14$^{\pm 1.68}$	78.39$^{\pm 1.29}$	94.50$^{\pm 0.40}$	**68.13**

Table 3: Unsupervised OOD detection performance on audio classification. We compare results from AP-OOD, Mahalanobis (Lee et al., 2018; Ren et al., 2023), KNN (Sun et al., 2022), Deep SVDD (Ruff et al., 2018), MSP (Hendrycks & Gimpel, 2016), and EBO (Liu et al., 2020b) trained on MIMII-DG (Dohi et al., 2022) as the ID data set. ↓ indicates "lower is better" and ↑ "higher is better". All values in %. We estimate standard deviations across five independent training runs.

	Mahalanobis	KNN	Deep SVDD	MSP	EBO	AP-OOD (Ours)
AUROC ↑	64.96$^{\pm 0.002}$	81.21$^{\pm 0.000}$	53.48$^{\pm 1.930}$	88.05$^{\pm 0.000}$	90.75$^{\pm 0.000}$	**92.86$^{\pm 0.746}$**
FPR95 ↓	84.39$^{\pm 0.011}$	57.11$^{\pm 0.000}$	89.44$^{\pm 1.689}$	36.43$^{\pm 0.000}$	61.86$^{\pm 0.000}$	**22.35$^{\pm 2.388}$**

In the audio task, the network achieves an accuracy of 97.6% on the primary classification task. Table 3 presents the results of the unsupervised OOD detection methods AP-OOD, Mahalanobis (Lee et al., 2018), KNN (Sun et al., 2022), and Deep SVDD (Ruff et al., 2018). Additionally, we compare AP-OOD to 2 methods for classifiers, Maximum Softmax Probability (MSP; Hendrycks & Gimpel, 2016) and Energy-based OOD Detection (EBO; Liu et al., 2020b). In contrast to the other methods, MSP and EBO do not apply to transformer tokens, making them unsuitable for summarization and translation tasks. The results show that AP-OOD improves the FPR95 metric from 36.43% (MSP) to 22.35%.

We evaluate the runtime performance of AP-OOD by measuring the inference time of single batches on the summarization task. We find that while AP-OOD is slower than the Mahalanobis baseline, it is still substantially faster than a forward pass through the transformer encoder. Because AP-OOD rejects a larger portion of OOD examples, the avoided generation time can effectively offset the AP-OOD's computational overhead relative to the baseline. For more details on the runtime behavior, we refer to Appendix D.9.

5 Related Work

OOD detection. Some authors (e.g., Bishop, 1994; Roth et al., 2022; Yang et al., 2022) distinguish between anomalies, outliers, and novelties. These distinctions reflect different goals within applications (Ruff et al., 2021). For example, when an anomaly is found, it will usually be removed from the training pipeline. However, when a novelty is found, it should be studied. We focus on detecting samples that are not part of the training distribution and consider sample categorization as a downstream task. OOD detection methods can be categorized into three groups: Post-hoc, training-time, and OE methods. In the post-hoc approach, one employs statistics obtained from a classifier. Perhaps the most well-known approach is the maximum softmax probability (MSP; Hendrycks & Gimpel, 2016). A wide range of post-hoc OOD detection approaches have been proposed to address the shortcomings of MSP (e.g., Lee et al., 2018; Hendrycks et al., 2019a; Liu et al., 2020a; Sun et al., 2021; 2022; Wang et al., 2022; Zhang et al., 2023b; Djurisic et al., 2023; Liu et al., 2023; Xu et al., 2024; Guo et al., 2025). A commonly used post-hoc method is the Mahalanobis distance (e.g., Lee et al., 2018; Sehwag et al., 2021; Ren et al., 2023). Recently, Müller & Hein (2025) proposed feature normalization to improve Mahalanobis-based OOD detection, and Guo et al. (2025) show that the Mahalanobis distance benefits from dynamically adjusting the prior geometry in response to new data. In contrast to post-hoc methods, training-time methods modify the training process of the encoder (e.g., Hendrycks et al., 2019c; Sehwag et al., 2021; Du et al., 2022; Hendrycks et al., 2022; Ming et al., 2023; Tao et al., 2023; Lu et al., 2024). Finally, the group of OE methods incorporates AUX data in the training process (e.g., Hendrycks et al., 2019b; Liu et al., 2020a; Ming et al., 2022; Zhang et al., 2023a; Wang et al., 2023; Zhu et al., 2023; Jiang et al., 2024; Hofmann et al., 2024).

OOD detection and natural language. Most of the aforementioned OOD detection approaches target vision tasks, and many of them require a classification model as the encoder ϕ. Applying these vision-based OOD methods to text is not straightforward due to the sequence-dependent nature of natural language (e.g., in autoregressive language generation). OOD detection specifically tailored for natural language is still underexplored. Ren et al. (2023) propose the log-model perplexity of a generated sequence $\boldsymbol{y}$ as a simple baseline for OOD detection on autoregressive language modeling tasks: $-\frac{1}{L}\sum_{l=1}^{L}\log p_\theta(y_l|\boldsymbol{y}_{<l},\boldsymbol{x})$. However, they show experimentally that model perplexity is inherently limited. Because of these shortcomings, Ren et al. (2023) propose embedding-based OOD detection methods for text data. Relatively few other works have explored OOD detection for generative language modeling. Notable applications include translation (e.g., Xiao et al., 2020; Malinin et al., 2021; Ren et al., 2023), summarization (Ren et al., 2023), and mathematical reasoning (Wang et al., 2024). A related field is hallucination detection (e.g., Malinin & Gales, 2020; Farquhar et al., 2024; Du et al., 2024; Aichberger et al., 2025; Park et al., 2025). Unlike OOD detection (which flags inputs outside the training distribution), the goal of hallucination detection is to identify prompts a generative language model is unlikely to answer truthfully.

6 Limitations & Future Work

We would like to discuss two limitations that we found. First, the selection of the AUX data is crucial, since it determines the shape of the ID–OOD decision boundary. If the AUX distribution diverges from the OOD examples faced at inference, the induced boundary may not be aligned with the task. Second, it remains unclear how reliably the OOD detection performance on specific data sets can indicate the general ability to detect OOD examples, as a large portion of plausible OOD inputs remains untested. An interesting avenue for future work is to apply OOD detection methods to large language models (LLMs; e.g., Abdin et al., 2024; Dubey et al., 2024; Yang et al., 2025). While we demonstrate the applicability of AP-OOD on the decoder-only language modeling paradigm of LLMs (Appendix D.6), further challenges include proprietary training data, finding OOD data for training and evaluation given the breadth of the ID data, and the ambiguity of p_{ID} arising from complex training pipelines involving multiple phases (e.g., Wei et al., 2022; Ouyang et al., 2022).

7 Conclusion

We introduce AP-OOD: an approach for OOD detection for natural language that can learn in supervised and unsupervised settings. In contrast to previous methods, AP-OOD learns how to pool token-level information without the explicit need for AUX data. Our experiments show that when supplied with AUX data during training, the performance of AP-OOD improves as more AUX data is provided. We compare AP-OOD to five unsupervised and three supervised OOD detection methods. Overall, AP-OOD shows the best results.

Reproducibility Statement

To ensure reproducibility, we provide the source code of our implementation of AP-OOD in the unsupervised and supervised settings in the supplementary material. Detailed instructions on running the source code and reproducing the experiments are provided in the file `readme.md`. We provide information about data, the training process, and the hyperparameter selection in Section 3 and Appendix D.2.

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

Lukas Aichberger, Kajetan Schweighofer, Mykyta Ielanskyi, and Sepp Hochreiter. Improving uncertainty estimation through semantically diverse language generation. In *The Thirteenth International Conference on Learning Representations*, 2025.

Ghadi S Al Hajj, Aliaksandr Hubin, Chakravarthi Kanduri, Milena Pavlovic, Knut Dagestad Rand, Michael Widrich, Anne Schistad Solberg, Victor Greiff, Johan Pensar, Günter Klambauer, et al. Incorporating probabilistic domain knowledge into deep multiple instance learning. In *Forty-first International Conference on Machine Learning*, 2024.

Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple-instance learning. *Advances in neural information processing systems*, 15, 2002.

Andreas Auer, Martin Gauch, Daniel Klotz, and Sepp Hochreiter. Conformal prediction for time series with modern hopfield networks. *Advances in Neural Information Processing Systems*, 36:56027–56074, 2023.

Mikko Aulamo and Jörg Tiedemann. The OPUS resource repository: An open package for creating parallel corpora and machine translation services. In Mareike Hartmann and Barbara Plank (eds.), *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pp. 389–394, Turku, Finland, September–October 2019. Linköping University Electronic Press.

Dzmitry Bahdanau. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

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

Christopher M Bishop. Novelty detection and neural network validation. *IEE Proceedings-Vision, Image and Signal processing*, 141(4):217–222, 1994.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia Specia, and Marco Turchi. Findings of the 2015 workshop on statistical machine translation. In Ondřej Bojar, Rajan Chatterjee, Christian Federmann, Barry Haddow, Chris Hokamp, Matthias Huck, Varvara Logacheva, and Pavel Pecina (eds.), *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pp. 1–46, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

Thomas G Dietterich, Richard H Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial intelligence*, 89(1-2):31–71, 1997.

Andrija Djurisic, Nebojsa Bozanic, Arjun Ashok, and Rosanne Liu. Extremely simple activation shaping for out-of-distribution detection. In *The Eleventh International Conference on Learning Representations*, 2023.

Kota Dohi, Tomoya Nishida, Harsh Purohit, Ryo Tanabe, Takashi Endo, Masaaki Yamamoto, Yuki Nikaido, and Yohei Kawaguchi. MIMII DG: Sound dataset for malfunctioning industrial machine investigation and inspection for domain generalization task. In *Proceedings of the 7th Detection and Classification of Acoustic Scenes and Events 2022 Workshop (DCASE2022)*, Nancy, France, November 2022.

Xuefeng Du, Zhaoning Wang, Mu Cai, and Yixuan Li. Vos: Learning what you don't know by virtual outlier synthesis. *arXiv preprint arXiv:2202.01197*, 2022.

Xuefeng Du, Chaowei Xiao, and Sharon Li. Haloscope: Harnessing unlabeled llm generations for hallucination detection. *Advances in Neural Information Processing Systems*, 37: 102948–102972, 2024.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630, 2024.

Andreas Fürst, Elisabeth Rumetshofer, Johannes Lehner, Viet T Tran, Fei Tang, Hubert Ramsauer, David Kreil, Michael Kopp, Günter Klambauer, Angela Bitto, et al. CLOOB: Modern Hopfield networks with InfoLOOB outperform clip. *Advances in neural information processing systems*, 35:20450–20468, 2022.

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

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. *arXiv preprint arXiv:1911.12237*, 2019.

Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.

Max Grusky, Mor Naaman, and Yoav Artzi. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. *arXiv preprint arXiv:1804.11283*, 2018.

Kaiyu Guo, Zijian Wang, Tan Pan, Brian C Lovell, and Mahsa Baktashmotlagh. Improving out-of-distribution detection via dynamic covariance calibration. In *Forty-second International Conference on Machine Learning*, 2025.

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. *arXiv preprint arXiv:1812.04606*, 2018.

Dan Hendrycks, Steven Basart, Mantas Mazeika, Andy Zou, Joe Kwon, Mohammadreza Mostajabi, Jacob Steinhardt, and Dawn Song. Scaling out-of-distribution detection for real-world settings. *arXiv preprint arXiv:1911.11132*, 2019a.

Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *International Conference on Learning Representations*, 2019b. URL https://openreview.net/forum?id=HyxCxhRcY7.

Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. *Advances in neural information processing systems*, 32, 2019c.

Dan Hendrycks, Andy Zou, Mantas Mazeika, Leonard Tang, Bo Li, Dawn Song, and Jacob Steinhardt. Pixmix: Dreamlike pictures comprehensively improve safety measures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16783–16792, 2022.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28, 2015.

Claus Hofmann, Simon Schmid, Bernhard Lehner, Daniel Klotz, and Sepp Hochreiter. Energy-based hopfield boosting for out-of-distribution detection. *arXiv preprint arXiv:2405.08766*, 2024.

J. J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences*, 81(10): 3088–3092, 1984. doi: 10.1073/pnas.81.10.3088.

Po-Yao Huang, Hu Xu, Juncheng Li, Alexei Baevski, Michael Auli, Wojciech Galuba, Florian Metze, and Christoph Feichtenhofer. Masked autoencoders that listen. In *NeurIPS*, 2022.

Rui Huang, Andrew Geng, and Yixuan Li. On the importance of gradients for detecting distributional shifts in the wild. *Advances in Neural Information Processing Systems*, 34: 677–689, 2021.

Maximilian Ilse, Jakub Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *International conference on machine learning*, pp. 2127–2136. PMLR, 2018.

Viktor Ivanov, Maurizio Scaramuzza, and Richard C Wilson. Deep temporal semi-supervised one-class classification for gnss radio frequency interference detection. *The Journal of Navigation*, 77(1):59–81, 2024.

Wenyu Jiang, Hao Cheng, MingCai Chen, Chongjun Wang, and Hongxin Wei. DOS: Diverse outlier sampling for out-of-distribution detection. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=iriEqxFB4y.

Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. Abstractive summarization of reddit posts with multi-level memory networks. *arXiv preprint arXiv:1811.00783*, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30, pp. 6402–6413, 2017.

Lisa Langnickel, Johannes Darms, Katharina Heldt, Denise Ducks, and Juliane Fluck. Continuous development of the semantic search engine preVIEW: from COVID-19 to long COVID. *Database*, 2022, 07 2022. ISSN 1758-0463. doi: 10.1093/database/baac048. URL https://doi.org/10.1093/database/baac048. baac048.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31, 2018.

Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 21464–21475. Curran Associates, Inc., 2020a. URL https://proceedings.neurips.cc/paper/2020/file/f5496252609c43eb8a3d147ab9b9c006-Paper.pdf.

Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in neural information processing systems*, 33:21464–21475, 2020b.

Xixi Liu, Yaroslava Lochman, and Christopher Zach. Gen: Pushing the limits of softmax-based out-of-distribution detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23946–23955, 2023.

Philipp Liznerski, Lukas Ruff, Robert A Vandermeulen, Billy Joe Franks, Klaus-Robert Müller, and Marius Kloft. Exposing outlier exposure: What can be learned from few, one, and zero outlier images. *arXiv preprint arXiv:2205.11474*, 2022.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Lefteris Loukas, Manos Fergadiotis, Ion Androutsopoulos, and Prodromos Malakasiotis. EDGAR-CORPUS: Billions of tokens make the world go round. In *Proceedings of the Third Workshop on Economics and Natural Language Processing*, pp. 13–18, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.econlp-1.2.

Haodong Lu, Dong Gong, Shuo Wang, Jason Xue, Lina Yao, and Kristen Moore. Learning with mixture of prototypes for out-of-distribution detection. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=uNkKaD3MCs.

Ian Magnusson, Akshita Bhagia, Valentin Hofmann, Luca Soldaini, Ananya Harsh Jha, Oyvind Tafjord, Dustin Schwenk, Evan Walsh, Yanai Elazar, Kyle Lo, et al. Paloma: A benchmark for evaluating language model fit. *Advances in Neural Information Processing Systems*, 37:64338–64376, 2024.

Andrey Malinin and Mark Gales. Uncertainty estimation in autoregressive structured prediction. *arXiv preprint arXiv:2002.07650*, 2020.

Andrey Malinin, Neil Band, German Chesnokov, Yarin Gal, Mark JF Gales, Alexey Noskov, Andrey Ploskonosov, Liudmila Prokhorenkova, Ivan Provilkov, Vatsal Raina, et al. Shifts: A dataset of real distributional shift across multiple large-scale tasks. *arXiv preprint arXiv:2107.07455*, 2021.

Oded Maron and Tomás Lozano-Pérez. A framework for multiple-instance learning. *Advances in neural information processing systems*, 10, 1997.

Yifei Ming, Ying Fan, and Yixuan Li. POEM: Out-of-distribution detection with posterior sampling. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15650–15665. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/ming22a.html.

Yifei Ming, Yiyou Sun, Ousmane Dia, and Yixuan Li. How to exploit hyperspherical embeddings for out-of-distribution detection? In *The Eleventh International Conference on Learning Representations*, 2023.

Maximilian Müller and Matthias Hein. Mahalanobis++: Improving ood detection via feature normalization. In *Forty-second International Conference on Machine Learning*, 2025.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*, 2018.

Tomoya Nishida, Noboru Harada, Daisuke Niizumi, Davide Albertini, Roberto Sannino, Simone Pradolini, Filippo Augusti, Keisuke Imoto, Kota Dohi, Harsh Purohit, et al. Description and discussion on dcase 2024 challenge task 2: First-shot unsupervised anomalous sound detection for machine condition monitoring. *arXiv e-prints*, pp. arXiv–2406, 2024.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Fabian Paischer, Thomas Adler, Vihang Patil, Angela Bitto-Nemling, Markus Holzleitner, Sebastian Lehner, Hamid Eghbal-Zadeh, and Sepp Hochreiter. History compression via language models in reinforcement learning. In *International Conference on Machine Learning*, pp. 17156–17185. PMLR, 2022.

Seongheon Park, Xuefeng Du, Min-Hsuan Yeh, Haobo Wang, and Yixuan Li. Steer llm latents for hallucination detection. *arXiv preprint arXiv:2503.01917*, 2025.

Hezhe Qiao, Qingsong Wen, Xiaoli Li, Ee-Peng Lim, and Guansong Pang. Generative semi-supervised graph anomaly detection. *Advances in neural information processing systems*, 37:4660–4688, 2024.

H. Ramsauer, B. Schäfl, J. Lehner, P. Seidl, M. Widrich, L. Gruber, M. Holzleitner, M. Pavlović, G. K. Sandve, V. Greiff, D. Kreil, M. Kopp, G. Klambauer, J. Brandstetter, and S. Hochreiter. Hopfield networks is all you need. In *9th International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=tL89RnzIiCd.

Jie Ren, Jiaming Luo, Yao Zhao, Kundan Krishna, Mohammad Saleh, Balaji Lakshminarayanan, and Peter J Liu. Out-of-distribution detection and selective generation for conditional language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=kJUS5nD0vPB.

Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14318–14328, 2022.

Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International conference on machine learning*, pp. 4393–4402. PMLR, 2018.

Lukas Ruff, Robert A Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. Deep semi-supervised anomaly detection. *arXiv preprint arXiv:1906.02694*, 2019.

Lukas Ruff, Jacob R Kauffmann, Robert A Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5):756–795, 2021.

Ana Sanchez-Fernandez, Elisabeth Rumetshofer, Sepp Hochreiter, and Guenter Klambauer. CLOOME: a new search engine unlocks bioimaging databases for queries with chemical structures. *bioRxiv*, 2022.

Bernhard Schäfl, Lukas Gruber, Angela Bitto-Nemling, and Sepp Hochreiter. Hopular: Modern Hopfield networks for tabular data. *arXiv preprint arXiv:2206.00664*, 2022.

Johannes Schimunek, Philipp Seidl, Lukas Friedrich, Daniel Kuhn, Friedrich Rippmann, Sepp Hochreiter, and Günter Klambauer. Context-enriched molecule representations improve few-shot drug discovery. In *The Eleventh International Conference on Learning Representations*, 2023.

Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, 12, 1999.

Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*, 2017.

Vikash Sehwag, Mung Chiang, and Prateek Mittal. Ssd: A unified framework for self-supervised outlier detection. *arXiv preprint arXiv:2103.12051*, 2021.

Zhuchen Shao, Hao Bian, Yang Chen, Yifeng Wang, Jian Zhang, Xiangyang Ji, et al. Transmil: Transformer based correlated multiple instance learning for whole slide image classification. *Advances in neural information processing systems*, 34:2136–2147, 2021.

Yiyou Sun and Yixuan Li. Dice: Leveraging sparsification for out-of-distribution detection. In *European Conference on Computer Vision*, pp. 691–708. Springer, 2022.

Yiyou Sun, Chuan Guo, and Yixuan Li. React: Out-of-distribution detection with rectified activations. *Advances in Neural Information Processing Systems*, 34:144–157, 2021.

Yiyou Sun, Yifei Ming, Xiaojin Zhu, and Yixuan Li. Out-of-distribution detection with deep nearest neighbors. In *International Conference on Machine Learning*, pp. 20827–20840. PMLR, 2022.

Leitian Tao, Xuefeng Du, Xiaojin Zhu, and Yixuan Li. Non-parametric outlier synthesis. *arXiv preprint arXiv:2303.02966*, 2023.

David MJ Tax and Robert PW Duin. Support vector data description. *Machine learning*, 54 (1):45–66, 2004.

Jörg Tiedemann. Parallel data, tools and interfaces in opus. In *Lrec*, volume 2012, pp. 2214–2218. Citeseer, 2012.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. Curran Associates, Inc., 2017a.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017b.

Haoqi Wang, Zhizhong Li, Litong Feng, and Wayne Zhang. Vim: Out-of-distribution with virtual-logit matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4921–4930, 2022.

Qizhou Wang, Zhen Fang, Yonggang Zhang, Feng Liu, Yixuan Li, and Bo Han. Learning to augment distributions for out-of-distribution detection. In *NeurIPS*, 2023. URL https://openreview.net/forum?id=OtU6VvXJue.

Yiming Wang, Pei Zhang, Baosong Yang, Derek Wong, Zhuosheng Zhang, and Rui Wang. Embedding trajectory for out-of-distribution detection in mathematical reasoning. *Advances in Neural Information Processing Systems*, 37:42965–42999, 2024.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=gEZrGCozdqR.

M. Widrich, B. Schäfl, M. Pavlović, H. Ramsauer, L. Gruber, M. Holzleitner, J. Brandstetter, G. K. Sandve, V. Greiff, S. Hochreiter, and G. Klambauer. Modern Hopfield networks and attention for immune repertoire classification. *ArXiv*, 2007.13505, 2020a.

Michael Widrich, Bernhard Schäfl, Milena Pavlović, Hubert Ramsauer, Lukas Gruber, Markus Holzleitner, Johannes Brandstetter, Geir Kjetil Sandve, Victor Greiff, Sepp Hochreiter, et al. Modern hopfield networks and attention for immune repertoire classification. *Advances in neural information processing systems*, 33:18832–18845, 2020b.

Tim Z Xiao, Aidan N Gomez, and Yarin Gal. Wat zei je? detecting out-of-distribution translations with variational transformers. *arXiv preprint arXiv:2006.08344*, 2020.

Kai Xu, Rongyu Chen, Gianni Franchi, and Angela Yao. Scaling for training time and post-hoc out-of-distribution detection enhancement. In *The Twelfth International Conference on Learning Representations*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Jingkang Yang, Pengyun Wang, Dejian Zou, Zitang Zhou, Kunyuan Ding, Wenxuan Peng, Haoqi Wang, Guangyao Chen, Bo Li, Yiyou Sun, et al. Openood: Benchmarking generalized out-of-distribution detection. *Advances in Neural Information Processing Systems*, 35: 32598–32611, 2022.

Jinsung Yoon, Kihyuk Sohn, Chun-Liang Li, Sercan O Arik, and Tomas Pfister. Spade: Semi-supervised anomaly detection under distribution mismatch. *Transactions on Machine Learning Research*, 2023.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International conference on machine learning*, pp. 11328–11339. PMLR, 2020.

Jingyang Zhang, Nathan Inkawhich, Randolph Linderman, Yiran Chen, and Hai Li. Mixture outlier exposure: Towards out-of-distribution detection in fine-grained environments. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 5531–5540, January 2023a.

Jinsong Zhang, Qiang Fu, Xu Chen, Lun Du, Zelin Li, Gang Wang, Shi Han, Dongmei Zhang, et al. Out-of-distribution detection based on in-distribution data patterns memorization with modern Hopfield energy. In *The Eleventh International Conference on Learning Representations*, 2023b.

Jianing Zhu, Yu Geng, Jiangchao Yao, Tongliang Liu, Gang Niu, Masashi Sugiyama, and Bo Han. Diversified outlier exposure for out-of-distribution detection via informative extrapolation. *Advances in Neural Information Processing Systems*, 36, 2023.

Table of Contents

A Related Work

Continuous modern Hopfield networks. Modern Hopfield networks (MHNs) are energy-based associative memory networks. They advance conventional Hopfield networks (Hopfield, 1984) by introducing continuous queries and states and a new energy function. MHNs have exponential storage capacity, while retrieval is possible with a one-step update (Ramsauer et al., 2021). The update rule of MHNs coincides with attention as it is used in the Transformer (Vaswani et al., 2017a). Examples for successful applications of MHNs are Widrich et al. (2020a); Fürst et al. (2022); Sanchez-Fernandez et al. (2022); Paischer et al. (2022); Schäfl et al. (2022); Schimunek et al. (2023); Auer et al. (2023) and Hofmann et al. (2024).

Multiple instance learning (MIL). MIL (Dietterich et al., 1997; Maron & Lozano-Pérez, 1997; Andrews et al., 2002; Ilse et al., 2018) considers a classifier that maps a bag $\boldsymbol{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_S)$ of instances $\boldsymbol{z}_s$ to a bag-level label $Y \in \{0, 1\}$. MIL also assumes that individual labels $y_s \in \{0, 1\}$ exist for the instances, which remain unknown during training. By assumption, the bag-level label is positive once one of the instance-level labels is positive (and negative if all are instance-level labels negative), i.e., $Y := \max_s y_s$. Recent MIL methods use attention pooling (Ilse et al., 2018; Shao et al., 2021; Al Hajj et al., 2024) and modern Hopfield networks (Widrich et al., 2020b) to pool the features of the instances.

One-class classification (OCC). OCC (Schölkopf et al., 1999) is the problem of learning a decision boundary separating the ID and OOD regions while having access to examples from the ID data set only. One-Class SVM (Schölkopf et al., 2001) learns a maximum margin hyperplane in the feature space that separates the ID data from the origin. Support Vector Data Description (SVDD; Tax & Duin, 2004) learns a hypersphere which encapsulates the ID data. Most closely related to AP-OOD is Deep SVDD (Ruff et al., 2018). Deep SVDD learns an encoder $\psi(\cdot, \mathcal{W}) : \mathbb{R}^D \rightarrow \mathbb{R}^M$ by minimizing the volume of a data-enclosing hypersphere in the output space. Ruff et al. (2019) propose Deep SAD, an extension of Deep SVDD that makes use of AUX data during training. However, Liznerski et al. (2022) show that the effectiveness of this extension degrades with increasing dimensionality.

B Theoretical Notes

B.1 OOD Score Investigation

In the following, we show that

$$\min_{j\in\{1,\ldots,M\}} -d_j^2(\phi_{\text{enc}}(\boldsymbol{x}), \tilde{\boldsymbol{Z}}) + \log(||\boldsymbol{w}_j||_2^2) < 2\log(\epsilon) \ + \ \log(2\pi) \quad \implies \quad \boldsymbol{x} \in \mathbb{O}$$

whenever $z_j := \frac{\boldsymbol{w}_j^T}{||\boldsymbol{w}_j||_2}\bar{\boldsymbol{z}}_j$ is normally distributed with probability density function

$$\dot{p}_j(z_j) \ := \ \frac{||\boldsymbol{w}_j||_2}{\sqrt{2\pi}} \ \exp\left(-\frac{1}{2}(||\boldsymbol{w}_j||_2 \ z_j \ - \ \boldsymbol{w}_j^T\boldsymbol{\mu}_j)^2\right), \tag{16}$$

weight vectors $\boldsymbol{w}_j \in \mathbb{R}^D$, encoder $\phi_{\text{enc}} : \mathcal{X} \to \mathcal{Z}$, $\mathcal{Z} = \bigcup_{S\geq 1}\mathbb{R}^{D\times S}$, $\boldsymbol{Z} \in \mathcal{Z}$, $\tilde{\boldsymbol{Z}} \in \mathcal{Z}$, $\bar{\boldsymbol{z}}_j = \boldsymbol{Z}\boldsymbol{p}_j$, $\boldsymbol{\mu}_j = \tilde{\boldsymbol{Z}}\tilde{\boldsymbol{p}}_j$, $\boldsymbol{p}_j \in \Delta^S$ and $\tilde{\boldsymbol{p}}_j \in \Delta^{S'}$ with

$$\Delta^S := \big\{(p_1, \ldots, p_S) \in [0,1]^S \mid \sum_{i=1}^{S} p_i = 1\big\}.$$

Proof. Note that the ϕ_{enc}-pushforward density $p_{\phi_{\text{enc}}}$ of p_{ID} satisfies

$$p_{\phi_{\text{enc}}}(\boldsymbol{Z}) := \int_{\mathcal{X}} p_{\text{ID}}(\boldsymbol{x}) \ \delta(\phi_{\text{enc}}(\boldsymbol{x}) = \boldsymbol{Z}) \, \mathrm{d}p_{\text{ID}}(\boldsymbol{x}) \geq \ p_{\text{ID}}(\boldsymbol{x}).$$

Analogously, we get $\bar{p}_j(\bar{\boldsymbol{z}}_j) \geq p_{\phi_{\text{enc}}}(\boldsymbol{Z})$ for $\bar{\boldsymbol{z}}_j := \boldsymbol{Z}\boldsymbol{p}_j$ and $\dot{p}_j(z_j) \geq \bar{p}_j(\bar{\boldsymbol{z}}_j)$ for $z_j := \frac{\boldsymbol{w}_j^T}{||\boldsymbol{w}_j||_2}\bar{\boldsymbol{z}}_j$.

That is, for any $j \in \{1, \ldots, M\}$, we have that $p_{\text{ID}}(\boldsymbol{x}) \leq p_{\phi_{\text{enc}}}(\boldsymbol{Z}) \leq \bar{p}_j(\bar{\boldsymbol{z}}_j) \leq \dot{p}_j(z_j)$. As a consequence, for all $j \in \{1, \ldots, M\}$ it holds that $\dot{p}_j(z_j) < \epsilon \implies p_{\text{ID}}(\boldsymbol{x}) < \epsilon$. Moreover, the following equivalence holds:

$$\begin{aligned}
\dot{p}_j(z_j) \ &< \ \epsilon && \iff \\
\frac{||\boldsymbol{w}_j||_2}{\sqrt{2\pi}} \ \exp\left(-\frac{1}{2}(||\boldsymbol{w}_j||_2 \ z_j \ - \ \boldsymbol{w}_j^T\boldsymbol{\mu}_j)^2\right) \ &< \ \epsilon && \iff \\
\frac{||\boldsymbol{w}_j||_2}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(\boldsymbol{w}_j^T\bar{\boldsymbol{z}}_j \ - \ \boldsymbol{w}_j^T\boldsymbol{\mu}_j)^2\right) \ &< \ \epsilon && \iff \\
-\ (\boldsymbol{w}_j^T\bar{\boldsymbol{z}}_j \ - \ \boldsymbol{w}_j^T\boldsymbol{\mu}_j)^2 \ + \ \log(||\boldsymbol{w}_j||_2^2) \ &< \ 2\log(\epsilon) \ + \ \log(2\pi) && \qquad (17)
\end{aligned}$$

As a consequence, we have that $\boldsymbol{x} \in \mathbb{O}$, if Equation (17) is satisfied for any $j \in \{1, \ldots, M\}$. □

B.2 Mahalanobis Decomposition

We assume the D weight vectors $\boldsymbol{w}_j$ are linearly independent. First, we start from the directional decomposition and show the relation to the Mahalanobis distance.

$$d_{\text{Maha}}^2(\bar{\boldsymbol{z}}, \boldsymbol{\mu}) \ = \ \sum_{j=1}^{D}\left(\boldsymbol{w}_j^T\bar{\boldsymbol{z}} - \ \boldsymbol{w}_j^T\boldsymbol{\mu}\right)^2 \tag{18}$$

$$= (\bar{\boldsymbol{z}} \ - \ \boldsymbol{\mu})^T\left(\sum_{i=1}^{D}\boldsymbol{w}_j\boldsymbol{w}_j^T\right)(\bar{\boldsymbol{z}} \ - \boldsymbol{\mu}) \tag{19}$$

$$= (\bar{\boldsymbol{z}} \ - \ \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\bar{\boldsymbol{z}} \ - \boldsymbol{\mu}). \tag{20}$$

Because the weight vectors are linearly independent, $\boldsymbol{\Sigma}^{-1}$ has full rank. Next, we go in the opposite direction and show that the eigenvectors $\boldsymbol{V} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_D)$ and eigenvalues $\boldsymbol{D} = \text{diag}(\lambda_1, \ldots, \lambda_D)$ of a $\boldsymbol{\Sigma}$ can be used to define corresponding $\boldsymbol{w}_j$.

$$
\begin{aligned}
d^2_{\text{Maha}}(\bar{\boldsymbol{z}}, \boldsymbol{\mu}) &= (\bar{\boldsymbol{z}} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\bar{\boldsymbol{z}} - \boldsymbol{\mu}) && (21)\\
&= (\bar{\boldsymbol{z}} - \boldsymbol{\mu})^T \boldsymbol{V}^T \boldsymbol{D}^{-1} \boldsymbol{V} (\bar{\boldsymbol{z}} - \boldsymbol{\mu}) && (22)\\
&= \left(\sqrt{\boldsymbol{D}^{-1}}\boldsymbol{V}\bar{\boldsymbol{z}} - \sqrt{\boldsymbol{D}^{-1}}\boldsymbol{V}\boldsymbol{\mu}\right)^T \left(\sqrt{\boldsymbol{D}^{-1}}\boldsymbol{V}\bar{\boldsymbol{z}} - \sqrt{\boldsymbol{D}^{-1}}\boldsymbol{V}\boldsymbol{\mu}\right) && (23)\\
&= \sum_{j=1}^{D} (\boldsymbol{w}_j^T \bar{\boldsymbol{z}} - \boldsymbol{w}_j^T \boldsymbol{\mu})^2, && (24)
\end{aligned}
$$

where $\boldsymbol{w}_j = \sqrt{\lambda_j^{-1}}\boldsymbol{v}_j$, $\boldsymbol{\Sigma} = \boldsymbol{V}^T\boldsymbol{D}\boldsymbol{V}$, and $\boldsymbol{\Sigma}^{-1} = \boldsymbol{V}^T\boldsymbol{D}^{-1}\boldsymbol{V}$.

The relation between the Mahalanobis distance and the directional decomposition is as follows:

1. Any linearly independent sequence $\boldsymbol{w}_1, \dots, \boldsymbol{w}_D$ induces a positive definite matrix $\boldsymbol{\Sigma}^{-1} := \sum_{j=1}^{D} \boldsymbol{w}_j \boldsymbol{w}_j^\top$, and hence a Mahalanobis distance satisfying

$$
\sum_{j=1}^{D} (\boldsymbol{w}_j^\top \bar{\boldsymbol{z}} - \boldsymbol{w}_j^\top \boldsymbol{\mu})^2 = (\bar{\boldsymbol{z}} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\bar{\boldsymbol{z}} - \boldsymbol{\mu}). \tag{25}
$$

2. Conversely, any full-rank covariance matrix $\boldsymbol{\Sigma}$ admits a set of linearly independent vectors $\boldsymbol{w}_1, \dots, \boldsymbol{w}_D$ such that $\boldsymbol{\Sigma}^{-1} = \sum_{j=1}^{D} \boldsymbol{w}_j \boldsymbol{w}_j^\top$, and therefore Equation (25) holds.

Thus, our decomposition and the Mahalanobis form represent the same quadratic form; the eigen-decomposition is only one possible choice of $\boldsymbol{w}_j$.

B.3 AP-OOD Reduces to Mahalanobis Distance with Mean Pooling for $\beta = 0$

In this section, we show that as $\beta = 0$ and $M = D$, $d^2(\boldsymbol{Z}, \tilde{\boldsymbol{Z}})$ reduces to the Mahalanobis distance with mean pooling as used by Ren et al. (2023). To arrive at the result, we assume uniform sequence lengths.

$$
\text{softmax}(0 \cdot \boldsymbol{Z}^T \boldsymbol{w})_s = \frac{\exp(0 \cdot \boldsymbol{z}_s^T \boldsymbol{w})}{\sum_{s'=1}^{S} \exp(0 \cdot \boldsymbol{z}_{s'}^T \boldsymbol{w})} = \frac{1}{S}, \tag{26}
$$

$$
\bar{\boldsymbol{z}} = \text{AttPool}_0(\boldsymbol{Z}, \boldsymbol{w}) = \boldsymbol{Z}\text{softmax}(0 \cdot \boldsymbol{Z}^T \boldsymbol{w}) = \frac{1}{S}\sum_{s=1}^{S} \boldsymbol{z}_s, \tag{27}
$$

$$
\boldsymbol{\mu} = \text{AttPool}_0(\tilde{\boldsymbol{Z}}, \boldsymbol{w}) = \tilde{\boldsymbol{Z}}\text{softmax}(0 \cdot \tilde{\boldsymbol{Z}}^T \boldsymbol{w}) = \frac{1}{SN}\sum_{i=1}^{N}\sum_{s=1}^{S} \boldsymbol{z}_{is} = \frac{1}{N}\sum_{i=1}^{N} \bar{\boldsymbol{z}}_i, \tag{28}
$$

where we use the concatenated sequence $\tilde{\boldsymbol{Z}} = (\boldsymbol{Z}_1 \| \cdots \| \boldsymbol{Z}_N)$, and the sequence representations $\boldsymbol{Z}_i = \phi(\boldsymbol{x}_i) = (\boldsymbol{z}_{i1}, \dots, \boldsymbol{z}_{iS}) \in \mathbb{R}^{D \times S}$. The squared distance of AP-OOD reduces to

$$
\begin{aligned}
d^2(\boldsymbol{Z}, \tilde{\boldsymbol{Z}}) &= \sum_{j=1}^{M} \left(\boldsymbol{w}_j^T \boldsymbol{Z}\text{softmax}(\beta\, \boldsymbol{Z}^T \boldsymbol{w}_j) - \boldsymbol{w}_j^T \tilde{\boldsymbol{Z}}\text{softmax}(\beta\, \tilde{\boldsymbol{Z}}^T \boldsymbol{w}_j)\right)^2 && (29)\\
&= \sum_{j=1}^{D} (\boldsymbol{w}_j^T \bar{\boldsymbol{z}} - \boldsymbol{w}_j^T \boldsymbol{\mu})^2 = d^2_{\text{Maha}}(\bar{\boldsymbol{z}}, \boldsymbol{\mu}). && (30)
\end{aligned}
$$

C Additional Algorithmic Details

C.1 Mini-Batch Attention Pooling

In this section, we describe the process of performing attention pooling over a long sequence $\tilde{\boldsymbol{Z}}$ that is too large to fit into memory by dividing $\tilde{\boldsymbol{Z}}$ into smaller mini-batches of size $B \in \mathbb{N}$. For this, we need the log-sum-exponential (lse) function. We follow the notation from Ramsauer et al. (2021):

$$\operatorname{lse}(\beta, \boldsymbol{a}) \;=\; \beta^{-1} \log \left(\sum_{s=1}^{S} \exp(\beta a_s) \right) \tag{31}$$

The following algorithm computes $\boldsymbol{\mu} = \tilde{\boldsymbol{Z}} \operatorname{softmax}(\beta\, \tilde{\boldsymbol{Z}}^T \boldsymbol{w})$ for $\beta > 0$:

Algorithm 2 Attention pooling over a long sequence

```
Require: $\tilde{\boldsymbol{Z}} = (\tilde{\boldsymbol{z}}_1, \ldots, \tilde{\boldsymbol{z}}_S) \in \mathbb{R}^{D \times S}$, inverse temperature $\beta$, weight vector $\boldsymbol{w}$, batch size $B$
 1: $E \leftarrow -\infty$
 2: $\boldsymbol{\mu} \leftarrow \mathbf{0}$
 3: for $s \leftarrow 1$ to $S$ step $B$ do
 4:     Load mini-batch $\boldsymbol{B} \leftarrow (\tilde{\boldsymbol{z}}_s, \ldots, \tilde{\boldsymbol{z}}_{s+B})$
 5:     $E_B \leftarrow \operatorname{lse}(\beta, \boldsymbol{B}^T \boldsymbol{w})$
 6:     $\boldsymbol{p} \leftarrow \exp(\beta(\boldsymbol{B}^T \boldsymbol{w} - E_B))$
 7:     $\boldsymbol{\mu}_B \leftarrow \boldsymbol{B}\boldsymbol{p}$
 8:     $p_B \leftarrow \sigma(\beta(E_B - E))$
 9:     $\boldsymbol{\mu} \leftarrow p_B \boldsymbol{\mu}_B + (1 - p_B)\boldsymbol{\mu}$
10:     $E \leftarrow \beta^{-1} \log(\exp(\beta E_B) + \exp(\beta E))$
    return $\boldsymbol{\mu}$
```

C.2 AP-OOD in PyTorch/Einops-like Pseudocode

We detail the loss computation for AP-OOD with multiple queries per head (Equation (14)) using PyTorch/Einops-style pseudocode. Assuming a uniform sequence length S, Algorithm 3 demonstrates the computation via attention pooling over the sequences $\mathbf{Z}$. Alternatively, Algorithm 4 presents a mathematically equivalent formulation that applies attention pooling over the similarities $\mathbf{W}^T \mathbf{Z}$.

C.3 On the Difference Between Heads and Queries

We find that heads are learnt largely independently from one another while queries are not, which we experimentally verify as follows: We train AP-OOD using the SGD optimizer on the summarization task using (i) 1 head and (ii) 2 heads, where the initialization of one of the heads in (ii) is identical to the initialization of the head of (i). We find that after training for 500 steps, the weight vectors associated with the heads with shared initialization between (i) and (ii) remain identical. In contrast, when repeating this experiment by varying the number of queries, the weight vectors associated with the queries differ after training. Intuitively, adding additional heads will help the model discover more local minima in the parameter space (similar to Lakshminarayanan et al., 2017), while adding queries increases the capacity of each given head. The following observation supports this intuition: When testing different hyperparameter combinations for AP-OOD, we found that a large number of queries combined with a small number of heads leads to overfitting when training the model on small ID data sets (e.g., 10,000 sequences): In this case, the average distance of the ID training sequences is substantially smaller than the average distance of the ID validation sequences.

Algorithm 3 AP-OOD loss in PyTorch/Einops-like style using attention pooling over $\boldsymbol{Z}$

```
def attention_pooling(Zs, Ws):
    # Zs[N S D] - mini-batch of N sequences with length S
    # Ws[M T D] - weights of model with M heads and T queries

    # pairwise similarities between tokens and weights
    similarities = einsum(Zs, Ws, 'N S D, M T D -> N M S T')

    # softmax over query- and sequence dimensions
    probs = similarities.softmax(dim=(-2, -1)) #[N M S T]

    # pooling to form new sequences
    Z_bars = einsum(Zs, probs, 'N S D, N M S T ->N M T D')

    return Z_bars

def loss(Zs, Ws):
    # Zs[N S D] - mini-batch of N sequences with length S
    # Ws[M T D] - weights of model with M heads and T queries

    # attention pooling over individual sequence
    Z_bars = attention_pooling(Zs, Ws) #[N M T D]

    # attention pooling over all sequences
    Z_tilde = Zs.flatten(0, 1).unsqueeze(0) #[1, N*S, D]
    mus = attention_pooling(Z_tilde, Ws) #[1 M T D]

    # squared distance per head
    heads_Z = einsum(Z_bars, Ws, 'N M T D, M T D -> N M')
    heads_mu = einsum(mus, Ws, '1 M T D, M T D -> 1 M')
    ds_squared = (heads_Z - heads_mu)**2 #[N M]

    # regularized and loss
    regularizer = torch.log((Ws * Ws).sum(1, 2)) #[M]
    losses = torch.sum(ds_squared - regularizer, dim=1) #[N]
    loss = torch.mean(losses)

    return loss
```

D Experiments

D.1 Additional Details for the Toy Experiment

In the toy experiment in Figure 2, we modify the attention pooling process to use the negative squared Euclidean distance instead of the dot product similarity because the Euclidean distance is known to work better in low-dimensional spaces. Formally, the modified attention pooling process is:

$$\mathrm{AttPool}_{\beta}(\boldsymbol{Z}, \boldsymbol{w}) \ := \ \sum_{s=1}^{S} \boldsymbol{z}_s \ \frac{\exp(-\frac{\beta}{2}||\boldsymbol{z}_s \ - \ \boldsymbol{w}||_2^2)}{\sum_{s'=1}^{S} \exp(-\frac{\beta}{2}||\boldsymbol{z}_{s'} \ - \ \boldsymbol{w}||_2^2)}. \tag{32}$$

D.2 Hyperparameter Selection

To find the values for β, M, and T in the unsupervised setting, we perform a grid search using the values $\beta \in \{\frac{1}{\sqrt{D}}, 0.25, 0.5, 1, 2\}$ and $T \in \{1, 4, 16\}$. We select M such that the total number of parameters of AP-OOD equals the number of entries in $\boldsymbol{\Sigma}$ of the Mahalanobis

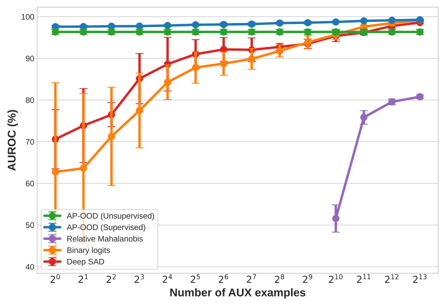
Algorithm 4 AP-OOD loss in PyTorch/Einops-like style using attention pooling over $\boldsymbol{W}^T\boldsymbol{Z}$

```
def attention_pooling(Zs, Ws):
    # Zs[N S D] - mini-batch of N sequences with length S
    # Ws[M T D] - weights of model with M heads and T queries

    # pairwise similarities between tokens and weights
    similarities = einsum(Zs, Ws, 'N S D, M T D -> N M S T')

    # softmax over query- and sequence dimensions
    probs = similarities.softmax(dim=(-2, -1)) #[N M S T]

    # pooling over similarities
    pooled = einsum(similarities, probs 'N M S T, N M S T -> N M')

    return pooled

def loss(Zs, Ws):
    # Zs[N S D] - mini-batch of N sequences with length S
    # Ws[M T D] - weights of model with M heads and T queries

    # attention pooling over individual sequence
    heads_Z = attention_pooling(Zs, Ws) #[N M]

    # attention pooling over all sequences
    Z_tilde = Zs.flatten(0, 1).unsqueeze(0) #[1, N*S, D]
    heads_mus = attention_pooling(Z_tilde, Ws) #[1 M]

    # squared distance per head
    ds_squared = (heads_Z - heads_mu)**2 #[N M]

    # regularizer and loss
    regularizer = torch.log((Ws * Ws).sum(1, 2)) #[M]
    losses = torch.sum(ds_squared - regularizer, dim=1) #[N]
    loss = torch.mean(losses)

    return loss
```

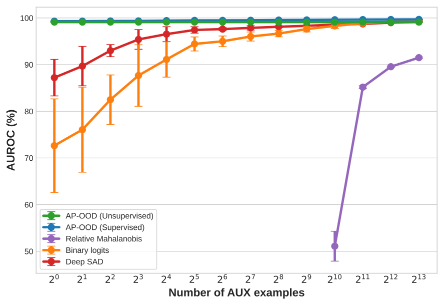
method, i.e., such that $MT = D$. We select the hyperparameter configuration by evaluating each resulting model on OOD detection using a validation split of the AUX data set (in the unsupervised setting, we use the AUX data set only for model selection, not for training the model), and we select the model with the highest AUROC. In the supervised setting, we follow the same procedure, and we additionally select $\lambda \in \{0.1, 1, 10\}$.

D.3 Supervised Experiments on Text Summarization

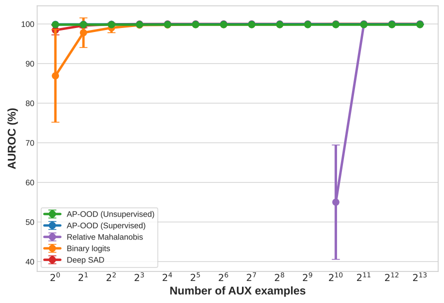
In the fully supervised setting, we train all methods on the embeddings of 100,000 ID examples and 10,000 AUX examples obtained from PEGASUS$_{\text{LARGE}}$ trained on text summarization using the XSUM data set. Table 4 shows that AP-OOD substantially improves fully supervised OOD detection results, improving the previously best mean FPR95 of 1.06% (binary logits) to 0.28% in the input OOD setting. Figure 5 shows the results for the semi-supervised setting when scaling the number of AUX examples on all OOD data sets for text summarization. We evaluate relative Mahalanobis only for $N' \geq 1024$, because $\boldsymbol{\Sigma}$ is not invertible when using fewer AUX examples. In contrast to Figure 3, Figure 5 also shows the results for Reddit TIFU and Samsum. On these two data sets, all evaluated methods except relative Mahalanobis achieve near-perfect OOD detection results for $N' \geq 8$.



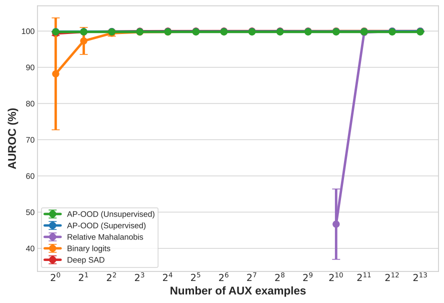
(a) CNN/DM



(b) Newsroom



(c) Reddit TIFU



(d) Samsum

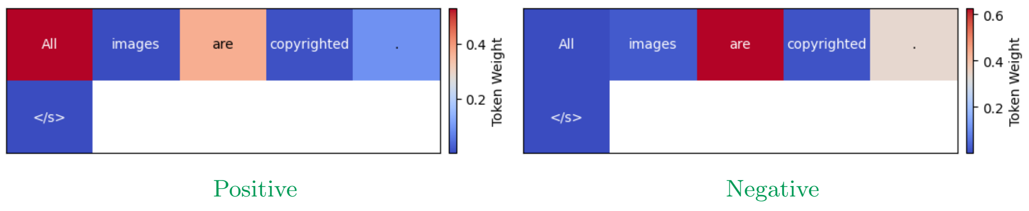
Figure 5: OOD detection performance on text summarization for all OOD data sets. We vary the number of AUX examples and compare results from AP-OOD, binary logits (Ren et al., 2023), relative Mahalanobis (Ren et al., 2023), and Deep SAD (Ruff et al., 2019).

Table 4: Supervised OOD detection performance on text summarization. We compare results from AP-OOD, binary logits (Ren et al., 2023), relative Mahalanobis (Ren et al., 2023), and Deep SAD (Ruff et al., 2019) on PEGASUS$_{\text{LARGE}}$ trained on XSUM as the ID data set. ↓ indicates "lower is better" and ↑ "higher is better". All values in %. We estimate standard deviations across five independent data set splits and training runs.

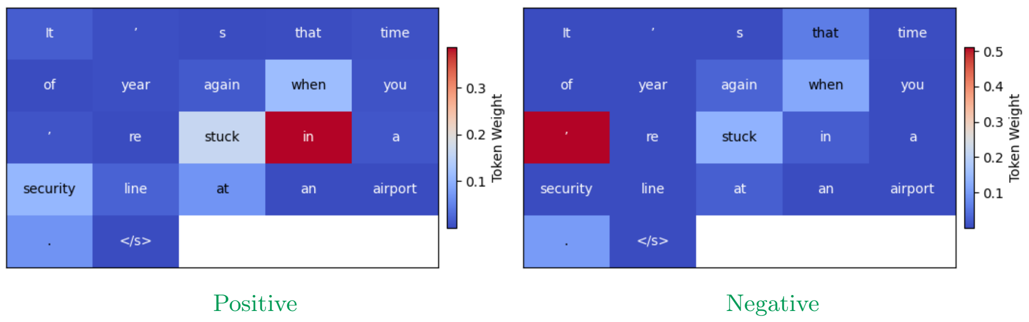
		CNN/DM	Newsroom	Reddit	Samsum	Mean
			Input OOD			
Binary logits	AUROC ↑	$\underline{99.43}^{\pm 0.11}$	$\underline{99.52}^{\pm 0.06}$	$100.00^{\pm 0.00}$	$99.99^{\pm 0.00}$	$\underline{99.73}$
	FPR95 ↓	$\underline{2.32}^{\pm 0.59}$	$\underline{1.93}^{\pm 0.17}$	$\mathbf{0.00}^{\pm \mathbf{0.00}}$	$0.01^{\pm 0.01}$	$\underline{1.06}$
Relative Mahalanobis	AUROC ↑	$81.28^{\pm 0.19}$	$91.85^{\pm 0.20}$	$99.96^{\pm 0.00}$	$99.98^{\pm 0.00}$	93.27
	FPR95 ↓	$62.92^{\pm 0.34}$	$28.22^{\pm 0.43}$	$0.00^{\pm 0.01}$	$0.01^{\pm 0.01}$	22.79
Deep SAD	AUROC ↑	$98.85^{\pm 0.17}$	$99.24^{\pm 0.07}$	$\underline{100.00}^{\pm 0.00}$	$\underline{100.00}^{\pm 0.00}$	99.52
	FPR95 ↓	$3.69^{\pm 0.81}$	$2.38^{\pm 0.16}$	$\mathbf{0.00}^{\pm \mathbf{0.00}}$	$\mathbf{0.00}^{\pm \mathbf{0.00}}$	1.52
AP-OOD (Ours)	AUROC ↑	$\mathbf{99.83}^{\pm \mathbf{0.18}}$	$\mathbf{99.71}^{\pm \mathbf{0.05}}$	$\mathbf{100.00}^{\pm \mathbf{0.00}}$	$\mathbf{100.00}^{\pm \mathbf{0.00}}$	**99.88**
	FPR95 ↓	$\mathbf{0.37}^{\pm \mathbf{0.51}}$	$\mathbf{0.76}^{\pm \mathbf{0.19}}$	$\mathbf{0.00}^{\pm \mathbf{0.00}}$	$\mathbf{0.00}^{\pm \mathbf{0.00}}$	**0.28**
			Output OOD			
Binary logits	AUROC ↑	$\underline{98.67}^{\pm 0.26}$	$99.49^{\pm 0.03}$	$99.99^{\pm 0.01}$	$99.94^{\pm 0.02}$	$\underline{99.52}$
	FPR95 ↓	$\underline{5.01}^{\pm 0.97}$	$1.77^{\pm 0.07}$	$0.00^{\pm 0.00}$	$0.09^{\pm 0.04}$	$\underline{1.72}$
Relative Mahalanobis	AUROC ↑	$93.58^{\pm 0.18}$	$97.41^{\pm 0.08}$	$99.82^{\pm 0.01}$	$99.54^{\pm 0.03}$	97.59
	FPR95 ↓	$24.32^{\pm 0.33}$	$8.54^{\pm 0.23}$	$0.04^{\pm 0.01}$	$1.00^{\pm 0.09}$	8.47
Deep SAD	AUROC ↑	$98.39^{\pm 0.23}$	$\underline{99.53}^{\pm 0.03}$	$\underline{100.00}^{\pm 0.00}$	$\underline{99.96}^{\pm 0.00}$	99.47
	FPR95 ↓	$6.00^{\pm 0.75}$	$\underline{1.66}^{\pm 0.14}$	$\mathbf{0.00}^{\pm \mathbf{0.00}}$	$\underline{0.07}^{\pm 0.03}$	1.93
AP-OOD (Ours)	AUROC ↑	$\mathbf{99.00}^{\pm \mathbf{0.13}}$	$\mathbf{99.59}^{\pm \mathbf{0.02}}$	$\mathbf{100.00}^{\pm \mathbf{0.00}}$	$\mathbf{99.98}^{\pm \mathbf{0.00}}$	**99.64**
	FPR95 ↓	$\mathbf{3.25}^{\pm \mathbf{0.42}}$	$\mathbf{1.24}^{\pm \mathbf{0.07}}$	$\mathbf{0.00}^{\pm \mathbf{0.00}}$	$\mathbf{0.01}^{\pm \mathbf{0.01}}$	**1.13**

D.4 Visualizing AP-OOD's Attention Maps on the Summarization Task

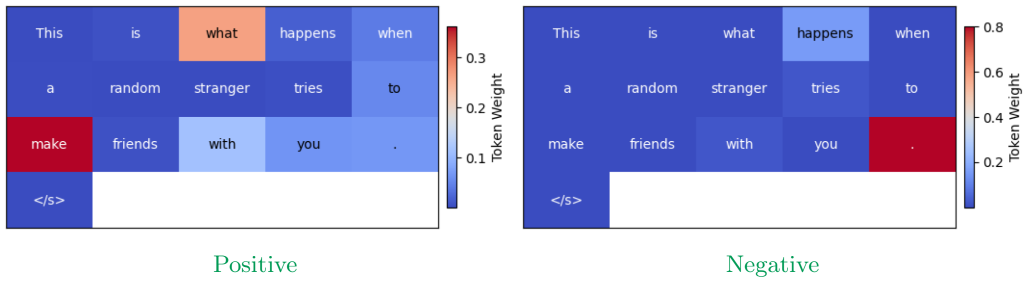
We analyze how the attention pooling process of AP-OOD allocates weight to individual tokens. We randomly select one sample from each of the four OOD data sets in the



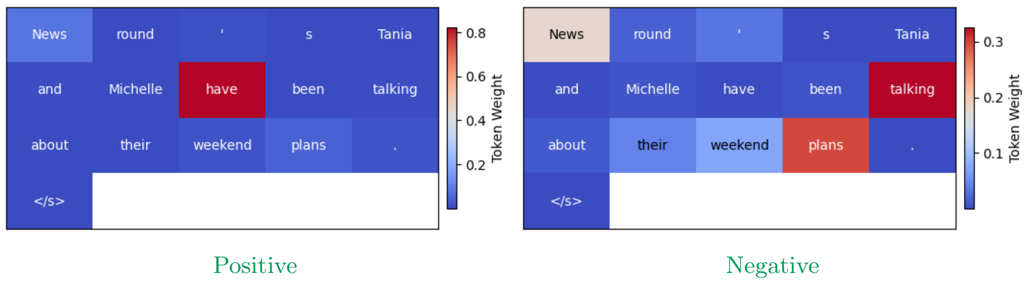
(a) CNN/DM



(b) Newsroom



(c) Reddit



(d) Samsum

Figure 6: AP-OOD's attention weights on randomly selected output sequences from OOD data sets on text summarization. For each sequence, we visualize the heads j with the highest deviation in the positive and negative direction of the $d_j(\boldsymbol{Z})$ before applying the square.

summarization benchmark. We then investigate the attention weights of a trained AP-OOD model over the generated output sequence. For each sample, we select the two heads with the largest deviations in the positive and in the negative directions before applying the square in the score function of AP-OOD. Figure 6 visualizes the token-wise attention scores of the selected heads. When manually examining the generated output sequences, we find it hard to attribute the "OODness" of individual sequences to a single token or to a small set of tokens. Therefore, it is difficult to interpret the attention scores for the individual heads. However, the results indicate that the different heads exhibit distinct attention patterns.

Table 5: Supervised OOD detection performance on English-to-French translation. We compare results from AP-OOD, binary logits, relative mahalanobis (Ren et al., 2023), and Deep SAD (Ruff et al., 2019) on a Transformer (base) trained on WMT15 En–Fr as the ID data set. ↓ indicates "lower is better" and ↑ "higher is better". All values in %. We estimate standard deviations across five independent data set splits and training runs.

		IT	Koran	Law	Medical	Subtitles	ndd2015	ndt2015	nt2014	Mean
					Input OOD					
Binary logits	AUROC ↑	$93.60^{\pm 0.34}$	$\underline{95.17^{\pm 0.05}}$	$54.29^{\pm 0.33}$	$70.47^{\pm 0.67}$	$90.53^{\pm 0.46}$	$89.91^{\pm 0.15}$	$\underline{89.80^{\pm 0.16}}$	$\underline{85.65^{\pm 0.06}}$	83.68
	FPR95 ↓	$28.58^{\pm 1.19}$	$\underline{34.91^{\pm 0.75}}$	$97.16^{\pm 0.06}$	$82.27^{\pm 0.64}$	$41.03^{\pm 0.96}$	$60.64^{\pm 0.41}$	$\underline{57.56^{\pm 0.58}}$	$\underline{75.78^{\pm 0.44}}$	59.74
Relative Mahalanobis	AUROC ↑	$92.82^{\pm 0.26}$	$93.31^{\pm 0.09}$	$43.07^{\pm 0.38}$	$\underline{74.40^{\pm 0.40}}$	$\mathbf{95.73^{\pm 0.21}}$	$89.33^{\pm 0.04}$	$88.88^{\pm 0.05}$	$82.06^{\pm 0.13}$	82.45
	FPR95 ↓	$\mathbf{19.27^{\pm 0.44}}$	$53.50^{\pm 0.68}$	$\mathbf{94.27^{\pm 0.24}}$	$\mathbf{67.63^{\pm 0.66}}$	$\mathbf{13.38^{\pm 0.28}}$	$\underline{59.06^{\pm 0.43}}$	$61.49^{\pm 0.37}$	$83.24^{\pm 0.14}$	$\underline{56.48}$
Deep SAD	AUROC ↑	$\underline{94.56^{\pm 0.13}}$	$94.77^{\pm 0.14}$	$\mathbf{57.44^{\pm 0.58}}$	$71.67^{\pm 0.27}$	$91.57^{\pm 0.21}$	$\underline{90.07^{\pm 0.16}}$	$89.47^{\pm 0.12}$	$84.42^{\pm 0.19}$	$\underline{84.25}$
	FPR95 ↓	$\underline{28.31^{\pm 0.62}}$	$40.77^{\pm 1.35}$	$97.10^{\pm 0.13}$	$83.74^{\pm 0.28}$	$41.15^{\pm 1.24}$	$61.54^{\pm 0.81}$	$62.11^{\pm 0.82}$	$79.33^{\pm 0.65}$	61.76
AP-OOD (Ours)	AUROC ↑	$\mathbf{94.97^{\pm 0.54}}$	$\mathbf{96.17^{\pm 0.35}}$	$\underline{56.82^{\pm 1.03}}$	$\mathbf{79.31^{\pm 0.99}}$	$\underline{95.03^{\pm 0.41}}$	$\mathbf{90.66^{\pm 0.39}}$	$\mathbf{90.73^{\pm 0.36}}$	$\mathbf{86.56^{\pm 0.36}}$	**86.28**
	FPR95 ↓	$29.93^{\pm 2.86}$	$\mathbf{26.04^{\pm 2.97}}$	$\underline{94.46^{\pm 0.83}}$	$\underline{79.06^{\pm 1.44}}$	$\underline{29.17^{\pm 2.32}}$	$\mathbf{56.34^{\pm 2.46}}$	$\mathbf{55.12^{\pm 1.47}}$	$\mathbf{69.75^{\pm 1.36}}$	**54.98**
					Output OOD					
Binary logits	AUROC ↑	$95.15^{\pm 0.06}$	$95.64^{\pm 0.17}$	$\underline{58.96^{\pm 0.79}}$	$74.70^{\pm 0.37}$	$92.79^{\pm 0.22}$	$\underline{90.32^{\pm 0.19}}$	$\underline{90.21^{\pm 0.16}}$	$\underline{85.73^{\pm 0.12}}$	85.44
	FPR95 ↓	$\underline{27.58^{\pm 0.44}}$	$30.49^{\pm 1.89}$	$96.36^{\pm 0.28}$	$82.09^{\pm 0.61}$	$39.08^{\pm 1.07}$	$\mathbf{57.36^{\pm 0.95}}$	$57.65^{\pm 0.68}$	$\underline{75.34^{\pm 0.41}}$	58.24
Relative Mahalanobis	AUROC ↑	$92.83^{\pm 0.18}$	$94.94^{\pm 0.14}$	$41.88^{\pm 0.42}$	$71.09^{\pm 0.27}$	$\underline{95.14^{\pm 0.16}}$	$88.86^{\pm 0.02}$	$87.83^{\pm 0.08}$	$82.59^{\pm 0.10}$	81.89
	FPR95 ↓	$28.72^{\pm 0.40}$	$36.30^{\pm 1.18}$	$\underline{95.54^{\pm 0.29}}$	$\mathbf{80.88^{\pm 0.20}}$	$\mathbf{20.42^{\pm 0.57}}$	$67.39^{\pm 0.52}$	$67.80^{\pm 0.48}$	$85.74^{\pm 0.20}$	60.35
Deep SAD	AUROC ↑	$\mathbf{95.88^{\pm 0.13}}$	$\underline{96.57^{\pm 0.21}}$	$56.47^{\pm 1.31}$	$\underline{76.35^{\pm 0.60}}$	$94.79^{\pm 0.12}$	$\mathbf{90.66^{\pm 0.11}}$	$\mathbf{90.40^{\pm 0.11}}$	$\mathbf{86.21^{\pm 0.18}}$	$\underline{85.92}$
	FPR95 ↓	$\mathbf{23.73^{\pm 0.47}}$	$\underline{21.38^{\pm 1.75}}$	$95.86^{\pm 0.38}$	$82.47^{\pm 0.52}$	$30.23^{\pm 0.82}$	$\underline{58.14^{\pm 1.45}}$	$\mathbf{57.37^{\pm 1.64}}$	$75.73^{\pm 0.23}$	$\underline{55.61}$
AP-OOD (Ours)	AUROC ↑	$\underline{95.82^{\pm 0.24}}$	$\mathbf{96.85^{\pm 0.24}}$	$\mathbf{59.22^{\pm 0.92}}$	$\mathbf{78.27^{\pm 1.67}}$	$\mathbf{95.78^{\pm 0.13}}$	$90.31^{\pm 0.33}$	$89.87^{\pm 0.35}$	$83.97^{\pm 0.90}$	**86.26**
	FPR95 ↓	$28.51^{\pm 1.44}$	$\mathbf{19.94^{\pm 1.78}}$	$\mathbf{93.65^{\pm 0.36}}$	$\underline{81.37^{\pm 0.56}}$	$\underline{26.96^{\pm 1.04}}$	$59.28^{\pm 1.36}$	$\underline{57.48^{\pm 1.09}}$	$\mathbf{73.64^{\pm 1.21}}$	**55.10**

D.5 Supervised Experiments on Translation

In the fully supervised setting, we train all methods on the embeddings of 100,000 ID embeddings and 100,000 AUX embeddings obtained from a Transformer (base) trained on WMT15 En–Fr translation. Table 5 shows that AP-OOD improves supervised OOD detection results w.r.t. the mean AUROC and mean FPR95 metrics.

D.6 Experiments on Decoder-Only Language Modeling

To verify the effectiveness of AP-OOD on the decoder-only language modeling paradigm used by LLMs, we conduct experiments on Pythia-160M (Biderman et al., 2023), a decoder-only language model trained on the Pile (Gao et al., 2020). We evaluate the discriminative power of AP-OOD trained in an unsupervised fashion on the 4Chan and Twitter subsets of Paloma (Magnusson et al., 2024), the EDGAR annual reports corpus (annual reports of public companies between 1993–2020; Loukas et al., 2021), Long-COVID related articles (Langnickel et al., 2022), and the MIMIC-III clinical corpus (Goldberger et al., 2000). In the decoder-only setting, we directly use the encoded representations of the input sequences and do not generate output sequences. Table 6 shows that AP-OOD improves unsupervised OOD detection w.r.t. the mean AUROC and mean FPR95 metrics.

D.7 OOD Score Comparison

We experimentally compare the min-based OOD score $s_{\min}(\boldsymbol{Z})$ and its upper bound $s(\boldsymbol{Z})$. For training, we use the loss from Equation (10) in both settings. The results in Table 7 show that $s(\boldsymbol{Z})$ achieves better OOD discrimination w.r.t. the mean AUROC and FPR95. While $s_{\min}(\boldsymbol{Z})$ roughly matches the OOD detection metrics of $s(\boldsymbol{Z})$ on CNN/DM for both input and output, $s_{\min}(\boldsymbol{Z})$ lags behind $s(\boldsymbol{Z})$ on the other OOD data sets.

D.8 Ablations

Beta sensitivity analysis. We evaluate AP-OOD when varying the hyperparameter β on the summarization task. We select β from $\{0, 1/\sqrt{D}, 0.25, 0.5, 1, 2\}$, and we leave the settings for M and T unchanged (i.e., they are identical to the settings used in Table 1). Table 8 shows that AP-OOD on text summarization is relatively insensitive to the selection of β inside the range $[0.25, 2]$ in the input and output settings.

Table 6: Unsupervised OOD detection performance on large-scale language modeling. We compare results from AP-OOD, Mahalanobis (Lee et al., 2018; Ren et al., 2023), KNN (Sun et al., 2022), DeepSVDD (Ruff et al., 2018), and model perplexity (Ren et al., 2023) on Pythia-160M trained on the Pile as the ID data set. ↓ indicates "lower is better" and ↑ "higher is better". All values in %. Standard deviations are estimated across five independent training runs.

		4Chan	Reports	Covid	Clinical	Twitter	Mean
		Input OOD					
Perplexity	AUROC ↑	$\underline{65.05}$	48.32	$\underline{89.51}$	$\underline{85.86}$	**99.22**	$\underline{77.59}$
	FPR95 ↓	**72.66**	**86.91**	$\underline{68.60}$	$\underline{65.22}$	$\underline{2.85}$	$\underline{59.25}$
Mahalanobis	AUROC ↑	35.27	54.72	75.50	75.67	97.86	67.81
	FPR95 ↓	92.93	$\underline{87.77}$	98.07	90.79	10.91	76.09
KNN	AUROC ↑	39.31	59.41	70.62	75.56	81.50	65.28
	FPR95 ↓	98.85	93.26	99.03	95.85	61.12	89.62
Deep SVDD	AUROC ↑	55.59	$\underline{64.06}$	72.54	73.44	81.08	69.34
	FPR95 ↓	$\underline{88.15}$	88.94	99.52	96.35	76.72	89.93
AP-OOD (Ours)	AUROC ↑	**87.97**	**68.09**	**91.79**	**86.44**	$\underline{99.08}$	**86.67**
	FPR95 ↓	88.34	91.47	**40.34**	**57.38**	**1.52**	**55.81**

Table 7: Unsupervised OOD detection performance on text summarization. We compare results from AP-OOD when using $s(\boldsymbol{Z})$ and $s_{\min}(\boldsymbol{Z})$, on PEGASUS$_{\text{LARGE}}$ trained on XSUM as the ID data set. ↓ indicates "lower is better" and ↑ "higher is better". All values in %. We estimate standard deviations across five independent dataset splits and training runs.

		CNN/DM	Newsroom	Reddit	Samsum	Mean
		Input OOD				
$s(\boldsymbol{Z})$	AUROC ↑	$\mathbf{96.13}^{\pm\mathbf{0.44}}$	$\mathbf{99.10}^{\pm\mathbf{0.08}}$	$\mathbf{99.91}^{\pm\mathbf{0.03}}$	$\mathbf{99.80}^{\pm\mathbf{0.04}}$	**98.74**
	FPR95 ↓	$19.51^{\pm 2.24}$	$\mathbf{4.11}^{\pm\mathbf{0.28}}$	$\mathbf{0.00}^{\pm\mathbf{0.01}}$	$\mathbf{0.04}^{\pm\mathbf{0.03}}$	**5.91**
$s_{\min}(\boldsymbol{Z})$	AUROC ↑	$96.08^{\pm 0.37}$	$97.48^{\pm 0.28}$	$99.71^{\pm 0.20}$	$97.67^{\pm 0.35}$	97.74
	FPR95 ↓	$\mathbf{18.78}^{\pm\mathbf{2.73}}$	$11.16^{\pm 1.21}$	$0.01^{\pm 0.01}$	$12.04^{\pm 3.04}$	10.50
		Output OOD				
$s(\boldsymbol{Z})$	AUROC ↑	$93.37^{\pm 0.54}$	$\mathbf{92.62}^{\pm\mathbf{0.67}}$	$\mathbf{98.04}^{\pm\mathbf{0.28}}$	$\mathbf{98.30}^{\pm\mathbf{0.11}}$	**95.59**
	FPR95 ↓	$\mathbf{23.12}^{\pm\mathbf{1.97}}$	$\mathbf{29.91}^{\pm\mathbf{2.93}}$	$\mathbf{6.34}^{\pm\mathbf{1.56}}$	$\mathbf{6.83}^{\pm\mathbf{0.64}}$	**16.55**
$s_{\min}(\boldsymbol{Z})$	AUROC ↑	$\mathbf{93.82}^{\pm\mathbf{1.56}}$	$88.30^{\pm 3.45}$	$95.94^{\pm 2.25}$	$90.13^{\pm 4.31}$	92.05
	FPR95 ↓	$26.60^{\pm 5.53}$	$38.26^{\pm 3.73}$	$18.49^{\pm 9.01}$	$36.71^{\pm 12.40}$	30.02

Number of heads M and queries T. We ablate on the number of heads M and the number of queries T of AP-OOD on the summarization task. For this ablation, we select $T \in \{1, 2, 4, 8, 16, 32, 64, 128, 512, 1024\}$ and we then select M such that the total number of parameters of AP-OOD equals the number of entries in $\boldsymbol{\Sigma}$ of the Mahalanobis method, i.e., such that $MT = D$. The results in Table 9 show that AP-OOD works best on the summarization task for both input and output when $M = 512$ and $T = 2$. Although the performance drops when decreasing M and increasing T, we find that AP-OOD is relatively insensitive to the number of heads and queries.

Dot product and Euclidean distance. We compare using the dot product and the negative squared Euclidean distance for the attention pooling in AP-OOD. For a formal definition of attention pooling with the negative squared Euclidean distance, we refer to Appendix D.1. Table 10 shows that using the dot product works substantially better. This result aligns with the well-established observation that measuring similarity using the dot product in high-dimensional spaces is more effective than using Euclidean distance.

D.9 Performance Measurements

We analyze the inference time of AP-OOD in comparison to the transformer backbone and other OOD detection methods. To avoid bottlenecks during data loading, we measure inference times on single batches and report the mean and standard deviation across 10

Table 8: Unsupervised OOD detection performance on text summarization. We compare results from AP-OOD trained on XSUM as the ID data set when varying β. ↓ indicates "lower is better" and ↑ "higher is better". All values in %. We estimate standard deviations across five independent dataset splits and training runs.

		CNN/DM	Newsroom	Reddit	Samsum	Mean
			Input OOD			
$\beta = 0$	AUROC ↑	$66.83^{\pm 0.44}$	$81.42^{\pm 0.27}$	$94.81^{\pm 0.32}$	$93.38^{\pm 0.20}$	84.11
	FPR95 ↓	$97.17^{\pm 0.10}$	$76.31^{\pm 0.35}$	$41.12^{\pm 3.42}$	$19.96^{\pm 0.84}$	58.64
$\beta = 0.25$	AUROC ↑	$\mathbf{97.76}^{\pm \mathbf{0.11}}$	$98.75^{\pm 0.07}$	$99.87^{\pm 0.06}$	$99.46^{\pm 0.09}$	**98.96**
	FPR95 ↓	$\mathbf{11.07}^{\pm \mathbf{0.74}}$	$4.75^{\pm 0.41}$	$\mathbf{0.00}^{\pm \mathbf{0.00}}$	$0.02^{\pm 0.02}$	**3.96**
$\beta = 0.5$	AUROC ↑	$96.13^{\pm 0.44}$	$\mathbf{99.10}^{\pm \mathbf{0.08}}$	$\mathbf{99.91}^{\pm \mathbf{0.03}}$	$99.80^{\pm 0.04}$	98.74
	FPR95 ↓	$19.51^{\pm 2.24}$	$\mathbf{4.11}^{\pm \mathbf{0.28}}$	$0.00^{\pm 0.01}$	$0.04^{\pm 0.03}$	5.91
$\beta = 1$	AUROC ↑	$91.36^{\pm 0.41}$	$98.77^{\pm 0.05}$	$99.75^{\pm 0.02}$	$99.83^{\pm 0.01}$	97.43
	FPR95 ↓	$38.78^{\pm 4.50}$	$4.94^{\pm 0.23}$	$0.02^{\pm 0.02}$	$\mathbf{0.00}^{\pm \mathbf{0.00}}$	10.94
$\beta = 2$	AUROC ↑	$84.29^{\pm 0.91}$	$97.58^{\pm 0.09}$	$99.52^{\pm 0.05}$	$99.76^{\pm 0.01}$	95.28
	FPR95 ↓	$63.31^{\pm 4.63}$	$9.14^{\pm 0.46}$	$0.12^{\pm 0.07}$	$0.05^{\pm 0.03}$	18.16
$\beta = 1/\sqrt{D}$	AUROC ↑	$89.09^{\pm 0.66}$	$90.59^{\pm 0.35}$	$99.59^{\pm 0.18}$	$\mathbf{99.87}^{\pm \mathbf{0.01}}$	94.79
	FPR95 ↓	$53.96^{\pm 3.30}$	$47.50^{\pm 1.83}$	$0.17^{\pm 0.18}$	$0.04^{\pm 0.02}$	25.42
			Output OOD			
$\beta = 0$	AUROC ↑	$77.67^{\pm 1.37}$	$85.10^{\pm 0.61}$	$84.12^{\pm 1.08}$	$91.70^{\pm 0.44}$	84.65
	FPR95 ↓	$82.07^{\pm 1.30}$	$69.32^{\pm 1.65}$	$57.30^{\pm 1.73}$	$29.37^{\pm 1.73}$	59.52
$\beta = 0.25$	AUROC ↑	$91.37^{\pm 0.64}$	$\mathbf{93.66}^{\pm \mathbf{0.13}}$	$94.79^{\pm 0.29}$	$96.56^{\pm 0.27}$	94.10
	FPR95 ↓	$43.03^{\pm 1.71}$	$34.70^{\pm 0.32}$	$38.38^{\pm 3.27}$	$18.61^{\pm 2.44}$	33.68
$\beta = 0.5$	AUROC ↑	$\mathbf{93.37}^{\pm \mathbf{0.54}}$	$92.62^{\pm 0.67}$	$\mathbf{98.04}^{\pm \mathbf{0.28}}$	$\mathbf{98.30}^{\pm \mathbf{0.11}}$	**95.59**
	FPR95 ↓	$\mathbf{23.12}^{\pm \mathbf{1.97}}$	$\mathbf{29.91}^{\pm \mathbf{2.93}}$	$\mathbf{6.34}^{\pm \mathbf{1.56}}$	$\mathbf{6.83}^{\pm \mathbf{0.64}}$	**16.55**
$\beta = 1$	AUROC ↑	$93.06^{\pm 0.57}$	$91.82^{\pm 0.71}$	$97.66^{\pm 0.33}$	$97.91^{\pm 0.22}$	95.11
	FPR95 ↓	$24.04^{\pm 1.95}$	$32.04^{\pm 2.97}$	$9.29^{\pm 1.71}$	$8.82^{\pm 1.42}$	18.55
$\beta = 2$	AUROC ↑	$93.25^{\pm 0.48}$	$91.98^{\pm 0.73}$	$97.57^{\pm 0.40}$	$97.97^{\pm 0.19}$	95.19
	FPR95 ↓	$23.69^{\pm 1.94}$	$31.23^{\pm 3.09}$	$10.06^{\pm 2.44}$	$8.37^{\pm 1.30}$	18.34
$\beta = 1/\sqrt{D}$	AUROC ↑	$54.67^{\pm 0.72}$	$80.59^{\pm 0.72}$	$94.12^{\pm 0.30}$	$94.93^{\pm 0.35}$	81.08
	FPR95 ↓	$92.40^{\pm 0.21}$	$65.83^{\pm 1.03}$	$30.04^{\pm 1.15}$	$27.20^{\pm 1.94}$	53.87

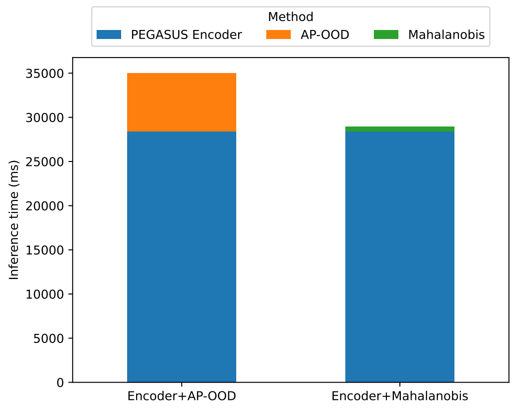


Figure 7: Comparing AP-OOD and the Mahalanobis method relative to the encoder inference time. The bars show the mean over ten batches.

batches. Our measurements only start after a warm-up phase of 5 batches. If not stated otherwise, the measurements were performed with a batch size of 32 and context length of 512 tokens. All measurements were performed on a single NVIDIA A100-40GB GPU.

Figure 8 compares the inference time of various OOD detection methods for different batch sizes. As expected AP-OOD has a strong linear relation to the batch size and is significantly slower than the reference models.

Although AP-OOD is slower than other methods like the Mahalanobis method, Figure 7 illustrates that it still takes less than 20 % of the combined inference time of the PEGASUS encoder and OOD detection method. We argue that the higher OOD detection rate mitigates the addition of overhead of AP-OOD since it allows skipping the substantially longer

Table 9: Unsupervised OOD detection performance on text summarization. We compare results from AP-OOD trained on XSUM as the ID data set when varying M and T. $\downarrow$ indicates "lower is better" and $\uparrow$ "higher is better". All values in %. We estimate standard deviations across five independent dataset splits and training runs.

		CNN/DM	Newsroom	Reddit	Samsum	Mean
			Input OOD			
$M = 1024 \quad T = 1$	AUROC $\uparrow$	$97.16^{\pm 0.22}$	$98.25^{\pm 0.11}$	$99.82^{\pm 0.01}$	$99.32^{\pm 0.03}$	98.64
	FPR95 $\downarrow$	$14.72^{\pm 0.83}$	$7.54^{\pm 0.62}$	$\underline{\mathbf{0.00}^{\pm\mathbf{0.00}}}$	$0.64^{\pm 0.11}$	5.72
$M = 512 \quad T = 2$	AUROC $\uparrow$	$\mathbf{97.98}^{\pm\mathbf{0.16}}$	$\mathbf{98.83}^{\pm\mathbf{0.07}}$	$\underline{99.87^{\pm 0.03}}$	$\mathbf{99.60}^{\pm\mathbf{0.04}}$	**99.07**
	FPR95 $\downarrow$	$\mathbf{9.77}^{\pm\mathbf{0.80}}$	$\mathbf{4.67}^{\pm\mathbf{0.30}}$	$\underline{\mathbf{0.00}^{\pm\mathbf{0.00}}}$	$\mathbf{0.02}^{\pm\mathbf{0.02}}$	**3.61**
$M = 256 \quad T = 4$	AUROC $\uparrow$	$\underline{97.76^{\pm 0.11}}$	$\underline{98.75^{\pm 0.07}}$	$\mathbf{99.87}^{\pm\mathbf{0.06}}$	$\underline{99.46^{\pm 0.09}}$	$\underline{98.96}$
	FPR95 $\downarrow$	$\underline{11.07^{\pm 0.74}}$	$\underline{4.75^{\pm 0.41}}$	$\underline{\mathbf{0.00}^{\pm\mathbf{0.00}}}$	$\underline{0.02^{\pm 0.02}}$	$\underline{3.96}$
$M = 128 \quad T = 8$	AUROC $\uparrow$	$97.53^{\pm 0.15}$	$98.49^{\pm 0.15}$	$99.83^{\pm 0.07}$	$99.14^{\pm 0.12}$	98.75
	FPR95 $\downarrow$	$12.48^{\pm 1.14}$	$5.94^{\pm 0.65}$	$\underline{\mathbf{0.00}^{\pm\mathbf{0.00}}}$	$0.25^{\pm 0.10}$	4.67
$M = 64 \quad T = 16$	AUROC $\uparrow$	$97.10^{\pm 0.09}$	$98.14^{\pm 0.16}$	$99.84^{\pm 0.07}$	$98.81^{\pm 0.16}$	98.47
	FPR95 $\downarrow$	$14.30^{\pm 0.77}$	$7.87^{\pm 0.86}$	$0.00^{\pm 0.00}$	$0.99^{\pm 0.50}$	5.79
$M = 32 \quad T = 32$	AUROC $\uparrow$	$96.84^{\pm 0.35}$	$97.78^{\pm 0.15}$	$99.83^{\pm 0.05}$	$98.56^{\pm 0.28}$	98.25
	FPR95 $\downarrow$	$14.97^{\pm 1.96}$	$10.18^{\pm 0.80}$	$0.01^{\pm 0.02}$	$2.53^{\pm 2.12}$	6.92
$M = 16 \quad T = 64$	AUROC $\uparrow$	$96.23^{\pm 0.45}$	$97.35^{\pm 0.24}$	$99.73^{\pm 0.11}$	$98.12^{\pm 0.24}$	97.86
	FPR95 $\downarrow$	$16.65^{\pm 1.99}$	$12.55^{\pm 1.15}$	$0.09^{\pm 0.20}$	$5.69^{\pm 1.87}$	8.75
$M = 8 \quad T = 128$	AUROC $\uparrow$	$95.56^{\pm 0.38}$	$96.47^{\pm 0.46}$	$99.67^{\pm 0.27}$	$97.44^{\pm 0.25}$	97.29
	FPR95 $\downarrow$	$18.16^{\pm 1.57}$	$16.34^{\pm 1.91}$	$0.52^{\pm 1.13}$	$11.29^{\pm 1.78}$	11.58
$M = 4 \quad T = 256$	AUROC $\uparrow$	$94.58^{\pm 0.67}$	$94.75^{\pm 0.52}$	$99.27^{\pm 0.86}$	$95.24^{\pm 0.25}$	95.96
	FPR95 $\downarrow$	$20.10^{\pm 2.32}$	$21.71^{\pm 2.30}$	$2.01^{\pm 4.09}$	$24.58^{\pm 1.83}$	17.10
$M = 2 \quad T = 512$	AUROC $\uparrow$	$93.17^{\pm 0.75}$	$91.87^{\pm 0.56}$	$98.43^{\pm 2.39}$	$89.87^{\pm 0.86}$	93.34
	FPR95 $\downarrow$	$22.86^{\pm 2.20}$	$27.09^{\pm 1.48}$	$4.95^{\pm 9.38}$	$39.75^{\pm 3.06}$	23.66
$M = 1 \quad T = 1024$	AUROC $\uparrow$	$90.90^{\pm 1.20}$	$88.10^{\pm 0.83}$	$96.68^{\pm 5.76}$	$81.41^{\pm 1.06}$	89.27
	FPR95 $\downarrow$	$27.14^{\pm 3.03}$	$32.64^{\pm 2.29}$	$9.03^{\pm 16.78}$	$52.73^{\pm 3.76}$	30.39
			Output OOD			
$M = 1024 \quad T = 1$	AUROC $\uparrow$	$92.47^{\pm 0.48}$	$94.17^{\pm 0.30}$	$98.36^{\pm 0.22}$	$97.77^{\pm 0.14}$	95.69
	FPR95 $\downarrow$	$39.11^{\pm 1.81}$	$34.69^{\pm 0.85}$	$3.11^{\pm 1.16}$	$12.59^{\pm 0.90}$	22.38
$M = 512 \quad T = 2$	AUROC $\uparrow$	$\mathbf{93.79}^{\pm\mathbf{0.25}}$	$\mathbf{95.85}^{\pm\mathbf{0.18}}$	$99.02^{\pm 0.20}$	$98.96^{\pm 0.06}$	**96.90**
	FPR95 $\downarrow$	$\mathbf{32.45}^{\pm\mathbf{1.29}}$	$\mathbf{20.10}^{\pm\mathbf{0.67}}$	$\underline{0.95^{\pm 0.66}}$	$\underline{2.77^{\pm 0.54}}$	**14.07**
$M = 256 \quad T = 4$	AUROC $\uparrow$	$\underline{93.35^{\pm 0.46}}$	$\underline{95.48^{\pm 0.28}}$	$\underline{99.19^{\pm 0.26}}$	$\mathbf{99.05}^{\pm\mathbf{0.06}}$	$\underline{96.77}$
	FPR95 $\downarrow$	$33.67^{\pm 2.77}$	$\underline{21.73^{\pm 0.82}}$	$\mathbf{0.86}^{\pm\mathbf{0.95}}$	$\mathbf{2.72}^{\pm\mathbf{0.52}}$	$\underline{14.75}$
$M = 128 \quad T = 8$	AUROC $\uparrow$	$93.24^{\pm 0.34}$	$95.27^{\pm 0.37}$	$\mathbf{99.21}^{\pm\mathbf{0.41}}$	$\underline{98.99^{\pm 0.04}}$	96.68
	FPR95 $\downarrow$	$\underline{32.84^{\pm 1.75}}$	$23.40^{\pm 1.53}$	$0.99^{\pm 1.56}$	$3.26^{\pm 0.42}$	15.12
$M = 64 \quad T = 16$	AUROC $\uparrow$	$92.95^{\pm 0.82}$	$94.92^{\pm 0.39}$	$99.11^{\pm 0.36}$	$98.89^{\pm 0.14}$	96.47
	FPR95 $\downarrow$	$34.08^{\pm 4.22}$	$25.53^{\pm 1.87}$	$1.48^{\pm 1.63}$	$4.10^{\pm 0.70}$	16.30
$M = 32 \quad T = 32$	AUROC $\uparrow$	$92.54^{\pm 0.61}$	$94.11^{\pm 0.47}$	$98.67^{\pm 0.73}$	$98.63^{\pm 0.41}$	95.99
	FPR95 $\downarrow$	$37.21^{\pm 3.76}$	$29.56^{\pm 2.71}$	$4.68^{\pm 4.39}$	$6.11^{\pm 2.55}$	19.39
$M = 16 \quad T = 64$	AUROC $\uparrow$	$91.26^{\pm 1.17}$	$92.62^{\pm 1.40}$	$97.99^{\pm 2.33}$	$98.58^{\pm 0.84}$	95.11
	FPR95 $\downarrow$	$41.96^{\pm 4.43}$	$35.78^{\pm 5.78}$	$8.75^{\pm 13.44}$	$6.19^{\pm 4.88}$	23.17
$M = 8 \quad T = 128$	AUROC $\uparrow$	$90.94^{\pm 1.97}$	$91.99^{\pm 1.88}$	$97.10^{\pm 2.54}$	$98.28^{\pm 0.80}$	94.58
	FPR95 $\downarrow$	$41.24^{\pm 8.00}$	$36.42^{\pm 7.58}$	$13.13^{\pm 13.35}$	$7.58^{\pm 3.85}$	24.59
$M = 4 \quad T = 256$	AUROC $\uparrow$	$89.62^{\pm 1.80}$	$90.35^{\pm 2.64}$	$95.91^{\pm 3.26}$	$97.73^{\pm 0.96}$	93.40
	FPR95 $\downarrow$	$47.52^{\pm 9.04}$	$41.77^{\pm 12.21}$	$18.53^{\pm 16.24}$	$10.02^{\pm 4.76}$	29.46
$M = 2 \quad T = 512$	AUROC $\uparrow$	$87.82^{\pm 2.50}$	$88.06^{\pm 1.29}$	$94.00^{\pm 3.38}$	$96.91^{\pm 1.26}$	91.70
	FPR95 $\downarrow$	$52.18^{\pm 9.71}$	$50.66^{\pm 5.51}$	$28.44^{\pm 17.40}$	$13.98^{\pm 6.18}$	36.31
$M = 1 \quad T = 1024$	AUROC $\uparrow$	$86.45^{\pm 1.86}$	$86.95^{\pm 1.79}$	$93.43^{\pm 2.35}$	$96.10^{\pm 1.59}$	90.73
	FPR95 $\downarrow$	$50.92^{\pm 8.94}$	$49.61^{\pm 6.70}$	$29.61^{\pm 8.37}$	$14.82^{\pm 3.62}$	36.24

generation time more often (see Table 11). The degree to which the overhead of AP-OOD is mitigated by skipping the decoder depends on the rate of detected OOD samples in future applications.

The heatmap in Figure 9 shows the inference time of AP-OOD for different selections of the hyperparameter number of heads M and number of queries T. While for small values of both parameters the inference time is constant, for larger parameters the inference time increases linearly with both parameters.

The inference time of the decoder of the transformer depends on the length of the longest output sequence of a batch. To obtain consistent measurements, we forced the decoder to always produce the same sequence length. Figure 10 illustrates the inference times of the PEGASUS transformer model. The plots of the first row cover the performance of the PEGASUS encoder only, while the second row shows the combined inference time of the encoder and decoder. In the left column, the plots indicate that the model inference time increases linearly with the batch size. Further, it is shown that the encoder takes about 10 % of the overall inference time. The right column shows the inference times for

Table 10: Unsupervised OOD detection performance on text summarization. We compare results from AP-OOD trained on XSUM as the ID data set when using the dot product and the Euclidean similarity. ↓ indicates "lower is better" and ↑ "higher is better". All values in %. We estimate standard deviations across five independent dataset splits and training runs.

		CNN/DM	Newsroom	Reddit	Samsum	Mean
Input OOD						
Dot product	AUROC ↑	**97.76**$^{\pm 0.11}$	**98.75**$^{\pm 0.07}$	**99.87**$^{\pm 0.06}$	**99.46**$^{\pm 0.09}$	**98.96**
	FPR95 ↓	**11.07**$^{\pm 0.74}$	**4.75**$^{\pm 0.41}$	**0.00**$^{\pm 0.00}$	**0.02**$^{\pm 0.02}$	**3.96**
Euclidean	AUROC ↑	74.22$^{\pm 0.65}$	84.43$^{\pm 0.23}$	97.06$^{\pm 0.41}$	98.30$^{\pm 0.23}$	<u>88.50</u>
	FPR95 ↓	90.20$^{\pm 0.37}$	74.08$^{\pm 1.04}$	15.27$^{\pm 5.30}$	7.17$^{\pm 1.94}$	<u>46.68</u>
Output OOD						
Dot product	AUROC ↑	**93.37**$^{\pm 0.54}$	**92.62**$^{\pm 0.65}$	**98.04**$^{\pm 0.29}$	**98.30**$^{\pm 0.11}$	**95.58**
	FPR95 ↓	**23.12**$^{\pm 1.98}$	**29.93**$^{\pm 2.89}$	**6.36**$^{\pm 1.60}$	**6.83**$^{\pm 0.64}$	**16.56**
Euclidean	AUROC ↑	87.67$^{\pm 0.74}$	88.17$^{\pm 1.80}$	96.50$^{\pm 0.57}$	91.28$^{\pm 1.79}$	<u>90.90</u>
	FPR95 ↓	65.62$^{\pm 3.90}$	66.04$^{\pm 4.38}$	22.34$^{\pm 5.36}$	53.89$^{\pm 7.80}$	<u>51.97</u>

Table 11: Inference times of OOD methods and PEGASUS transformer for a batch size of 32 samples, number of heads $M = 256$, number of queries $T = 4$, and a context length of $S = 512$ tokens. All values in milliseconds ms. We estimate the mean and standard deviation over ten batches.

AP-OOD	Mahalanobis	PEGASUS Encoder	PEGASUS Generation
6.58$^{\pm 0.095}$	0.52$^{\pm 0.146}$	28.43$^{\pm 2.513}$	34940.34$^{\pm 65.842}$

an increasing number of context tokens. The inference time of the transformer encoder and decoder increases quadratically with the context length.

E The Use of Large Language Models

When creating this paper, we utilized large language models (LLMs) to refine our writing, to identify related work, and for research ideation. When refining the writing using LLMs, we carefully review and verify LLM output to preserve sentence semantics. For related work, we confirm the soundness of papers suggested by the LLM, and for research ideation, we verify the factual accuracy of all statements.

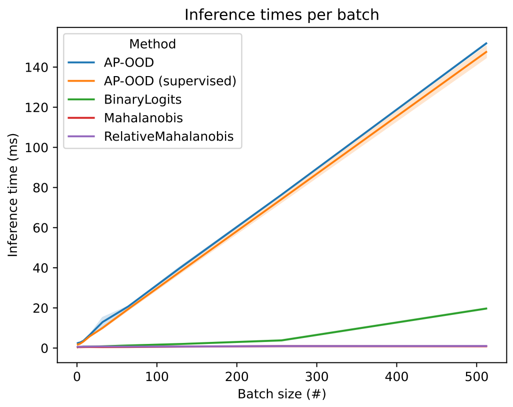


Figure 8: Comparison of various OOD detection methods for increasing batch sizes. We estimate the mean and standard deviation over ten batches.

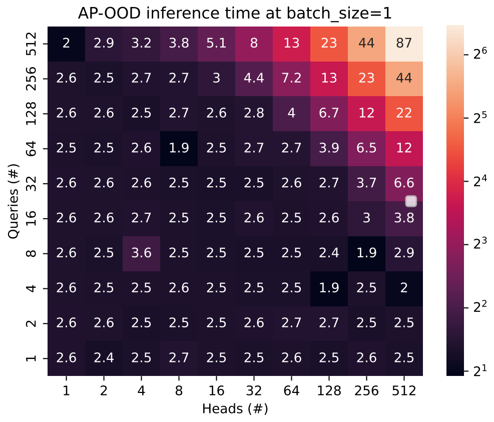


Figure 9: Inference times for AP-OOD over different numbers of heads M and queries T. We estimate the mean over ten batches.

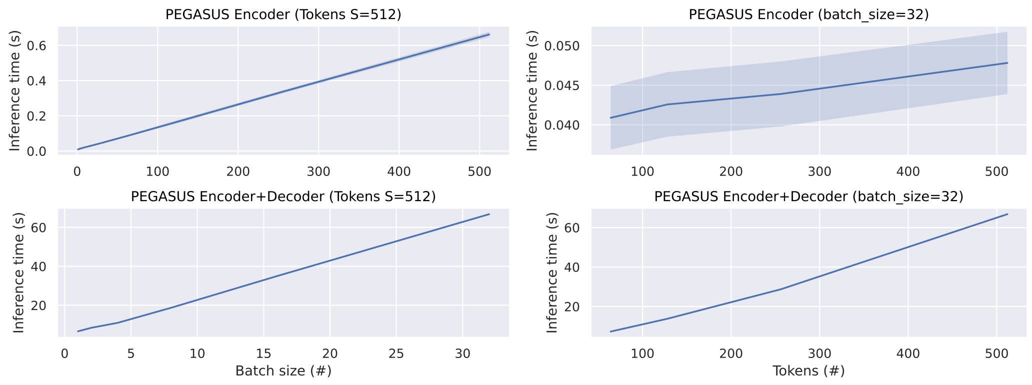


Figure 10: Inference times for the PEGASUS model for various batch sizes (left) and various numbers of input tokens (right). The top row illustrates the Encoder's performance, while the bottom row shows the combined performance of the Encoder and Decoder. We estimate the mean and standard deviation over ten batches.