Leveraging latent representations for efficient textual OOD detection

Lilian Marey
ENSAE
lilian.marey@ensae.fr

Lucas Saban
ENSAE
lucas.saban@ensae.fr

Abstract

With the democratization of Large-Language-Models through the release of products such as ChatGPT, the public is being more and more sensitive to flaws known for a long time in the community. An example of that is the overconfidence in the responses to Out-Of-Distribution (OOD) queries. Those are likely to be non-relevant and be detrimental to the public sentiment. However, this could be prevented by OOD detectors which are able to determine whether the model will be able to produce a satisfactory answer. We focus our work on out-of-the-bag detectors that are model-independent and do not leverage the model structure, but only an input's latent representations. We first reproduce the results of Colombo et al. and Guerreiro et al. on a restricted benchmark, then we argue that a better leverage of latent representations can lead to improved performances. To this extent, we introduce an exponential-based and an euclidean-distance-based methods to make our point. The code leading to the experimental results is available on GitHub.[1]

1 Introduction

As the use of large models in real-world applications continues to grow, ensuring their robustness and reliability becomes a critical concern. Two key challenges that arise in deploying large models into production are Out-of-Distribution (OOD) detection (Darrin et al., 2023a,b; Gomes et al.) and adversarial attack detection (Picot et al., 2023a,b). OOD detection refers to the ability to detect inputs that differ significantly from the data that the model was trained on, which can lead to erroneous predictions and other issues. Adversarial attack detection, on the other hand, involves detecting when an attacker deliberately manipulates the inputs to the model in order to cause it to make incorrect predictions. Both OOD detection and adversarial attack detection are crucial to ensuring that large models are able to function correctly in real-world scenarios, and are therefore important areas of research and development in the field of machine learning.

This study places its emphasis on OOD detection, an area that has been largely neglected in the context of textual data. Text classification models are built based on the assumption that the training and testing data follow the same probability distribution. After training a model, it can be evaluated on any dataset with the correct format. However, when the probability law of the test data is different from that of the training data, it results in incorrect predictions. A dataset that violates the probability law of the training set is referred to as out of distribution (OOD). We choose to based our work on (Colombo et al., 2022) which mainly work on the latent space of the multilayer neural networks

The paper aims to utilize the latent space representation of a neural network to detect OOD data. To achieve this, three databases are required: the training database used to train the network, a test database with the same probability distribution as the training database, and an OOD database that differs in distribution from the training database.

2 Related Work

At each output of a Transformer layer, a vector encodes the data. The average vector of the latent space representations is assigned to each data point, resulting in the latent training, test, and OOD distributions.

Several similarity metrics, such as Integrated Rank-Weighted Depth and Mahalanobis (Ghorbani, 2019), are utilized to compute similarity scores between the test and training distributions

[1] github.com/lilianmarey/nlp_ood_detection

and between the OOD and training distributions. The computed threshold value is then used to construct the OOD detector.

The Integrated Rank-Weighted Depth (D_{IRW}) is a metric which allows computing a distance between a vector x and a collection of vectors $(x_i)_{1\leq i\leq n}$ of the same distribution. We used the Monte-Carlo approximation[2] of D_{IRW} in order to compute the distance :

$$\tilde{D}_{IRW} = \frac{1}{n_{proj}} \sum_{k=1}^{n_{proj}} min(f(u_k), 1 - f(u_k))$$

where $(u_i)_{1\leq i\leq n_{proj}}$ are sampled vectors on the n-dimensional sphere and $f(u_k) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}(\langle u_k, x - x_i\rangle \leq 0)$

The point of (Colombo et al., 2022) is that using D_{IRW} depth on the average latent representations allows obtaining state-of-the-art results in OOD detection. In this paper, we will build a benchmark to evaluate the performance of these techniques through several metrics. Moreover, we will challenge the choice of using the average of the data representation in the latent space by proposing an alternative aggregation method putting more importance on the last layers of the neural network. We will also introduce a new method based on the Euclidean distance.

3 Experimental Protocol

As a base model, we consider a BERT model, fine-tuned on the IMDB database, thus for a binary classification task. We were able to import directly this pre-trained model from Hugging Face[3]. The network is composed of 12 "Bert Layer" of auto-encoders, with an output size of 768. We thus based our data representation on the average of 12 latent vectors of size 768.

We consider four other datasets as OOD datasets : sst2, flickr30K, WM16 (de-en) and 20newsgroup.

We use two metrics to evaluate the performances of the different OOD-Detectors. Following (Colombo et al., 2022), we use two threshold-invariant metrics, the Area Under the Receiver Operating Curve (AUROC) and the Area Under the Precision-Recall curve (AUPR).

$$\text{AUROC} = \int_0^1 \text{TPR}(FPR^{-1}(t)), \mathrm{d}t \qquad (1)$$

[2]We used the Python implementation of Staerman et al.

[3]huggingface.co/fabriceyhc/bert-base-uncased-imdb

where TPR is the true positive rate and FPR is the false positive rate.

$$\text{AUPR} = \int_0^1 \text{P}(r), \mathrm{d}r \qquad (2)$$

where P is the precision and r is the recall.

We will now introduce the different detectors and their philosophy we will be benchmarking.

3.1 OOD Detectors

The philosophy between the different detectors we are going to present is to detect anomalies and, as such, to build similarity measures between what *should be* and what *is*. There are two approaches : the first is based on the heuristic that OOD samples exhibit particularly *non-uniforms* logits or attentions and the second one is data-driven, comparing the logits, attentions or embeddings with the expected distribution of the train samples.

Let's first introduce the different notations we will use. *logits* refers to the logits returned by Bert for classification.The notation $\overline{embd}$ is the mean of different embeddings obtained after each Bert Layer. *softmax* refers to the function :

$$x \in \mathbb{R}^d \to \text{softmax}(x) := (\frac{e^{x_1}}{\sum_i e^{x_i}}, ..., \frac{e^{x_d}}{\sum_i e^{x_i}}) \in \mathbb{R}^d$$

LogSumExp is a soft surrogate of the max :

$$x \in \mathbb{R}^d \to \text{LogSumExp}(x) := \log(\sum_i e^{x_i}) \in \mathbb{R}$$

attentions refers to the attention weights obtained in the last layer of Bert (shape 768).

Finally, D_M is the Mahalanobis distance and D_{IRW} is the Integrated-Rank-Weighted distance introduced in (Colombo et al., 2022), whereas $\mathcal{W}_1$ corresponds to 1-Wassertein distance.

Non-uniformity Detectors

Maximum soft-probability (MSP)

$$s_{MSP}(x) = 1 - \max_{y\in\mathcal{Y}} [\text{softmax} \circ \text{logits}](x)$$

Energy (E)

$$s_E(x) = T \times [\text{LogSumExp} \circ \text{logits}](\frac{x}{T})$$

Wasserstein-to-uniform (W2U)Guerreiro et al.

$$s_{W2U}(x) = \mathcal{W}_1(\text{attentions}(x), \mathcal{U}_{|\text{attentions}|})$$

Data-driven Detectors

Mahalanobis (M)

$$s_M(x) = -D_M(\overline{\text{embd}}(x), \overline{\text{embd}}(X_{train})_{y=\hat{y}})$$

TRUSTED (IRW)

$$s_{IRW}(x) = -D_{IRW}(\overline{\text{embd}}(x), \overline{\text{embd}}(X_{train})_{y=\hat{y}})$$

Wasserstein-to-data (W2D) Guerreiro et al.

$$r_j(x) = \mathcal{W}_1(\text{attentions}(x), \text{attentions}(X_{train})^{(j)}_{y=\hat{y}})$$

$$s_{W2D}(x) = \frac{1}{k} \sum_{k \text{ smallest}} r_j(x)$$

Wasserstein-combo (WC) (Guerreiro et al., 2022) This measure is a combination of W2U and W2D in order to get the best of both. Given a train set, we first estimate the W2U scores, and define τ_u as the 98^{th} percentile. We define s_{WC} as :

$$\begin{aligned} s_{WC}(x) =& \mathbb{1}[s_{W2U}(x) > \tau_u] * s_{W2U}(x) \\ &+ \mathbb{1}[s_{W2U}(x) \leq \tau_u] * s_{W2D}(x) \end{aligned}$$

For computationally expensive methods such as Wass2Data, where we have to compute all the distances between the train set and the test set, we sampled uniformly a given percentage of the dataset to represent it. In fact, for a train set of n_{train}, dimension d and a test set of n_{test} samples, it represents $n_{train}n_{test}d$ operations. It can be of magnitude 10^{11} in certain cases. Hence, at each call, a new batch of 10% of the entire dataset is independently drawn. The method introduced in section 4 also uses random sampling.

3.2 Challenging the aggregation method

For reasons of simplicity and computational efficiency, (Colombo et al., 2022) makes the choice to aggregate the layers by averaging. This choice implicitly asserts that the data representation would be as important in the latent space at the beginning of the network as at the end. However, it can be noted that baseline methods and the literature in general tend to think that the last layers of the network capture information better than the beginning layers. We therefore propose to aggregate by putting more weight on the vectors at the end of the network.

Instead of averaging by $F_{PM}(x) = \frac{1}{L}\sum_{l=1}^{L} \phi_l(x)$, we propose the **exponential aggregation**

$$F_\alpha(x)_i = \frac{1}{L}\sum_{l=1}^{L} \exp(\alpha l \phi_l(x)_i)$$

with an appropriate alpha.

To evaluate this new aggregation, we take the paper pipeline and directly evaluate the performance of the OOD-detector. Still on the BERT network, we evaluate the detector by the Err metric defined in (Colombo et al., 2022), using histograms based on Mahalanobis and IRW similarities of IMDB (test) vs SST2 (OOD). One could argue that putting the embedding in the exponential could stack the negative values and so decrease representation space expressiveness. We tried putting it outside the exponential term and results were just not convincing, this is why we decided of keeping this aggregation method.

4 A new method : LiLO

Given a train in-distribution dataset $\mathcal{X}^{in}$ and a train OOD-dataset $\mathcal{X}^{out}$, we define the matrix $\mathcal{E}(X) = (\mathcal{E}_1(X), ..., \mathcal{E}_L(X))^T \in \mathbb{R}^{L \times d}$ where $\mathcal{E}_l(X) \in \mathbb{R}^d$ is the l^{th} d-dimensional embedding. Alongside, we define $\mathcal{E}_2(\cdot) := \mathcal{E}(\cdot)\mathcal{E}(\cdot)^T \in \mathbb{R}^{L^2}$ and $\overline{U}(X) = \frac{1}{|X|}\sum_{x \in X} U(x)$, with $X \subset \mathbb{R}^d$ and $U : \mathbb{R}^d \to \mathbb{R}^m, m \in \{L, L^2\}$

Also given a distance measure D, our goal is to find some α parameterizing a class of functions such that intra-dataset distances are minimized while the distances in/ood are maximized. A formulation of this problem can be :

$$\begin{aligned} \min_{\alpha, f} & \frac{1}{|\mathcal{X}^{in}|^2} \sum_{u_{in}, v_{in}} D(f_\alpha(u_{in}), f_\alpha(v_{in})) \\ &+ \frac{1}{|\mathcal{X}^{out}|^2} \sum_{u_{out}, v_{out}} D(f_\alpha(u_{out}), f_\alpha(v_{out}) \\ &- \frac{1}{|\mathcal{X}^{in}||\mathcal{X}^{out}|} \sum_{u_{in}, v_{out}} D(f_\alpha(u_{in}), f_\alpha(v_{out}) \end{aligned}$$

We studied the case where f_α is linear ($f_\alpha = \mathcal{E}^T\alpha$) and $D = ||\cdot||_2^2$. We will refer to this setup as linear-ℓ_2-opt (LiLO). We can reformulate the problem as (demonstration is available in Appendix B) :

$$\min_{\alpha \in \mathbb{R}^L} \alpha^T G \alpha$$

where :

$$\begin{aligned} G =& \overline{\mathcal{E}_2}(X^{in}) + \overline{\mathcal{E}_2}(X^{out}) \\ &- (\overline{\mathcal{E}}(X^{in}) - \overline{\mathcal{E}}(X^{out}))\left(\overline{\mathcal{E}}(X^{in}) - \overline{\mathcal{E}}(X^{out})\right)^T \end{aligned}$$

We can recognize an eigenvalue's problem as we can prove that $G \succ 0$ using convexity arguments.

The solution is α^* an eigenvector associated with $\lambda^* = \min Sp(G)$

A benchmark varying the OOD-Train set is available in Appendix D.

For example, in using imdb as in-ds, sst2 as ood-train and wmt16 as ood-test, we obtain the scores distribution displayed in Figure 1 and Figure 2 if we are aggregating using the mean or α^*

As displayed in Figure 3, we can see that the weights can be negative, and of different scales. The scale comparison is possible as we deal with normalized embeddings.

5 Results

5.1 Benchmark

The different score's distribution can be found in Appendix C.

Methods	AUROC	AUPR
Mahalanobis	0.964	0.943
IRW	0.964	0.942
LiLO	0.850	0.443
Wass2data	0.821	0.369
Wasscombo	0.817	0.363
Energy	0.700	0.198
MSP	0.696	0.950
Wass2unif	0.593	0.904

Table 1: Results for BERT (Averaged over OOD Datasets)

5.2 Exponential aggregation

The Err curves over α are presented in Appendix E.

Methods	F_{PM}	$F_{.001}$	$F_{.236}$	$F_{.239}$
Mahalanobis	8.99	9.07	**7.61**	7.64
IRW	9.02	9.33	7.70	**7.42**

Table 2: Err scores for exponential aggregation

6 Discussion/Conclusion

The Optimal-Transport inspired distances, described in (Guerreiro et al., 2022), seem to underperform in our benchmark. This is probably due to the fact that in the original paper, the hidden dimension is smaller. The task being Neural Machine Translation, the attention map is computed between tokens and therefore is of dimension of the order of 10. In our setup, the attention is computed on the embeddings, and therefore on dimension 768. The attention-focus used to distinguish OOD samples may not be discriminating enough to obtain more relevant results.

Indeed, we obtained the same result with models for NMT Task as can be seen in Figure 4. However, it is worth noticing that even with this drawback, the method outperforms some methods of the benchmark.

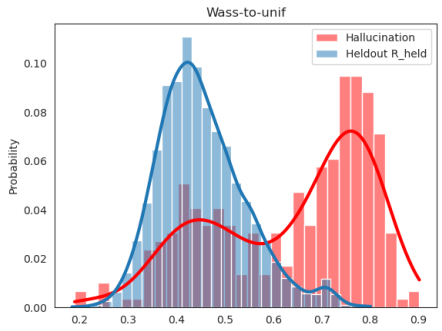


Figure 4: Wass2Unif with NMT backbone

Moreover, the performances obtained with a simple norm such as ℓ_2 suggest that optimizing the aggregation function of the embeddings might improve results as much as using more evolved distance measures. A pursuit of this paper could be to combine the linear aggregation obtained with LiLO with metrics such as IRW and the Mahalanobis distance, or to try to use kernels to explore non-linear embedding layers combinations.

About the exponential aggregation: even if the choice of the best α is expensive in time and space because it is necessary to store the latent vectors of all the layers for all the data, and to build a detector for each α to be tested, using this aggregation allows improving the results consequently. The next step could be to calculate the optimal α for several databases, in order to give a heuristic for the choice of its value.

To conclude, we add support to (Colombo et al., 2022) in the senses that out-of-the-bag OOD detector can be built using similarity measures. In our restricted benchmark (less OOD test sets than in the original paper), we did not find any significant performance gap between the Mahalanobis-based distance and the TRUSTED method. However, we were able to show that the aggregation method of embedding layers could be of great influence in improving discrimina-

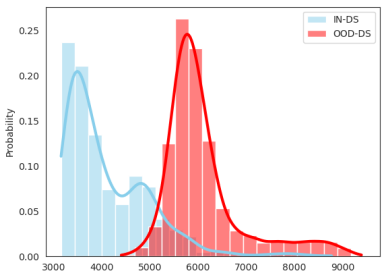


Figure 1: Using α^* as projection for ℓ_2 distance

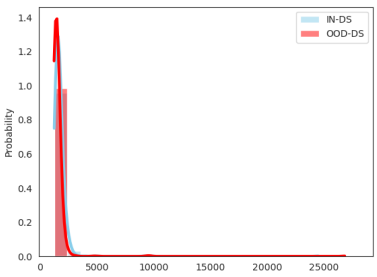


Figure 2: Using the mean as aggregation function for ℓ_2 distance leads to bad results

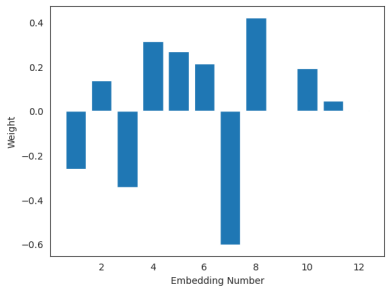


Figure 3: α obtained using LiLO on sst2

tive performances, even when trained against a particular OOD dataset (LiLO). We introduced an exponential-based aggregation function and a new method based on the Euclidean distance that demonstrated in relatively simple setups how could a well-designed aggregation function lead to competitive results. We hope that this would convince researchers to try to gain a deeper understanding of how information flows across layers, and thus paving the way for elegant OOD detectors that are able to leverage this structure.

References

Eduardo Dadalto Câmara Gomes, Pierre Colombo, Guillaume Staerman, Nathan Noiry, and Pablo Piantanida. A functional perspective on multi-layer out-of-distribution detection.

Dan Hendrycks and Kevin Gimpel. 2018. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. ArXiv:1610.02136 [cs].

Hamid Ghorbani. 2019. Mahalanobis distance and its application for detecting multivariate outliers. *Facta Universitatis, Series: Mathematics and Informatics*, pages 583–595.

Guillaume Staerman, Pavlo Mozharovskyi, and Stéphan Clémençon. 2021. Affine-invariant integrated rank-weighted depth: Definition, properties and finite sample analysis.

Pierre Colombo, Eduardo D. C. Gomes, Guillaume Staerman, Nathan Noiry, and Pablo Piantanida. 2022. Beyond Mahalanobis-Based Scores for Textual OOD Detection. ArXiv:2211.13527 [cs].

Nuno M. Guerreiro, Pierre Colombo, Pablo Piantanida, and André F. T. Martins. 2022. Optimal Transport for Unsupervised Hallucination Detection in Neural Machine Translation. ArXiv:2212.09631 [cs].

Marine Picot, Nathan Noiry, Pablo Piantanida, and Pierre Colombo. 2023a. Adversarial attack detection under realistic constraints.

Maxime Darrin, Pablo Piantanida, and Pierre Colombo. 2023a. Rainproof: An umbrella to shield text generators from out-of-distribution data. *arXiv preprint arXiv:2212.09171*.

Marine Picot, Guillaume Staerman, Federica Granese, Nathan Noiry, Francisco Messina, Pablo Piantanida, and Pierre Colombo. 2023b. A simple unsupervised data depth-based method to detect adversarial images.

Maxime Darrin, Guillaume Staerman, Eduardo Dadalto Câmara Gomes, Jackie CK Cheung, Pablo Piantanida, and Pierre Colombo. 2023b. Unsupervised layer-wise score aggregation for textual ood detection. *arXiv preprint arXiv:2302.09852*.

A Procedure explanation of (Hendrycks and Gimpel, 2018)

We explain here how we build OOD detectors using a soft classifier approach as introduced in (Hendrycks and Gimpel, 2018).

Let $\mathcal{D}_n = (X_i, Y_i)_{i\in[|1,n|]} \in (\mathcal{X} \times [|1, d|])^n$ and $\mathcal{F}$ a set of functions. The functions of is set are such that, for any $f \in \mathcal{F}$, $f : \mathcal{X} \to \Delta_d$ where Δ_d is the d-dimensional simplex.

The function $f_{\mathcal{D}_n}$ is a *soft*-classifier trained on a $\mathcal{D}_n$ whose natural *hard*-classifier is

$$F_{\mathcal{D}_n}(\cdot) = \arg\max_{i\in[|1,n|]} [f_{\mathcal{D}_n}(\cdot)]_i$$

From this, and given a threshold, $t \in (0, 1)$ we define a *correctly-classified* classifier C_{corr} and *misclassified* classifier C_{mis} such that :

$$C_{corr}(X_k)=\mathbb{1}(\max_i [f_{\mathcal{D}_n}(X_k)]_i \geq t)\mathbb{1}(F_{\mathcal{D}_n}(X_k)=Y_k)$$

and

$$C_{mis}(X_k)=\mathbb{1}(\max_i [f_{\mathcal{D}_n}(X_k)]_i \geq t)\mathbb{1}(F_{\mathcal{D}_n}(X_k)\neq Y_k)$$

Similarly, by changing the second indicator to $\mathbb{1}\left((X_k, Y_k) \in \mathcal{D}\right)$ or $\mathbb{1}\left((X_k, Y_k) \notin \mathcal{D}\right)$, we can define $\mathcal{C}_{in}$ and $\mathcal{C}_{out}$ as defined in the original paper.

Finally, by using the AUPR and AUROC metrics, which are threshold-independent, we have valid classifiers independently of the threshold t chosen.

B LiLO closed-form

If we consider the ℓ_2-norm and if f_α is such that

$$f_\alpha(\cdot) = \mathcal{E}^T(\cdot)\alpha$$

The quantity we are minimizing is $A + B - C$:

$$A := \frac{1}{|\mathcal{X}^{in}|^2} \sum_{u_{in}, v_{in}} ||\mathcal{E}^T(u_{in})\alpha - \mathcal{E}^T(v_{in})\alpha||_2^2$$

$$B := \frac{1}{|\mathcal{X}^{out}|^2} \sum_{u_{out}, v_{out}} ||\mathcal{E}^T(u_{out})\alpha - \mathcal{E}^T(v_{out})\alpha||_2^2$$

$$C := \frac{1}{|\mathcal{X}^{in}||\mathcal{X}^{out}|} \sum_{u_{in}, v_{out}} ||\mathcal{E}^T(u_{in})\alpha - \mathcal{E}^T(v_{out})\alpha||_2^2$$

We can compute separately each terms :

$$\begin{aligned}
A =& \frac{1}{|\mathcal{X}^{in}|^2} \sum_{u_{in}, v_{in}} \alpha^T(\mathcal{E}(u_{in}) - \mathcal{E}(v_{in}))(\mathcal{E}(u_{in}) - \mathcal{E}(v_{in}))^T\alpha \\
=& \alpha^T\Big(\frac{1}{|\mathcal{X}^{in}|^2} \sum_{u_{in}, v_{in}} (\mathcal{E}(u_{in})\mathcal{E}(u_{in})^T + \mathcal{E}(v_{in}))\mathcal{E}(v_{in}))^T \\
& - \mathcal{E}(u_{in})\mathcal{E}(v_{in})^T - \mathcal{E}(v_{in})\mathcal{E}(u_{in})^T\Big)\alpha \\
=& 2\alpha^T\Big(\overline{\mathcal{E}_2}(X^{in}) - \overline{\mathcal{E}}(X^{in})\overline{\mathcal{E}}(X^{in})^T\Big)\alpha
\end{aligned}$$

Similarly, we have :

$$\begin{aligned}
B =& 2\alpha^T\Big(\overline{\mathcal{E}_2}(X^{out}) - \overline{\mathcal{E}}(X^{out})\overline{\mathcal{E}}(X^{out})^T\Big)\alpha \\
C =& \alpha^T\Big(\overline{\mathcal{E}_2}(X^{in}) + \overline{\mathcal{E}_2}(X^{out}) - \overline{\mathcal{E}}(X^{in})\overline{\mathcal{E}}(X^{out})^T \\
& - \overline{\mathcal{E}}(X^{out})\overline{\mathcal{E}}(X^{in})^T\Big)\alpha
\end{aligned}$$

Hence,

$$A + B - C = \alpha^T G \alpha$$

where :

$$\begin{aligned}
G =& \overline{\mathcal{E}_2}(X^{in}) + \overline{\mathcal{E}_2}(X^{out}) \\
& - \left(\overline{\mathcal{E}}(X^{in}) - \overline{\mathcal{E}}(X^{out})\right)\left(\overline{\mathcal{E}}(X^{in}) - \overline{\mathcal{E}}(X^{out})\right)^T
\end{aligned}$$

□

One could see that G seems to look like a covariance matrix between the embeddings of each dataset. It would be interesting to study the different properties of this matrix, and the forms it takes for other norms, such as the ℓ_1 for example, provided they exist.

C Benchmark

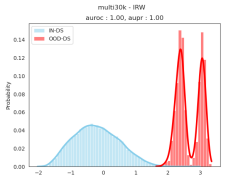
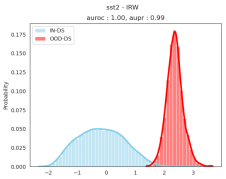
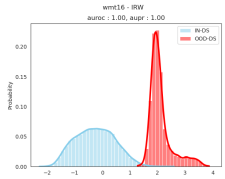
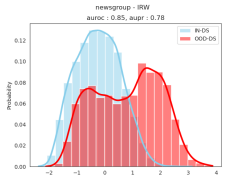
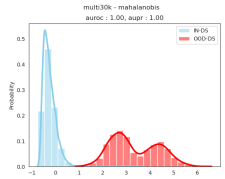
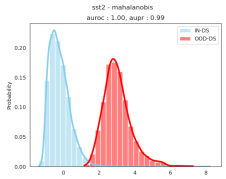
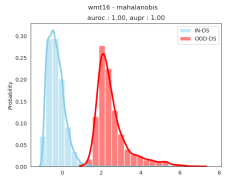
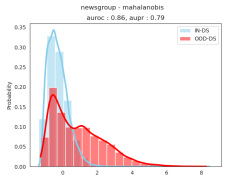
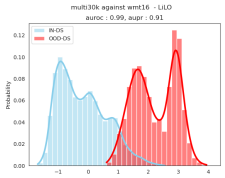
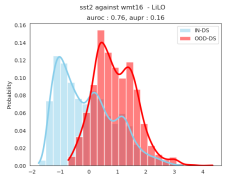
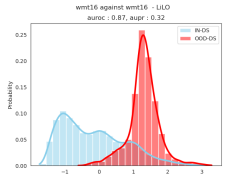
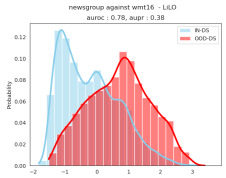
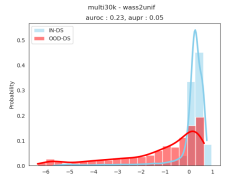
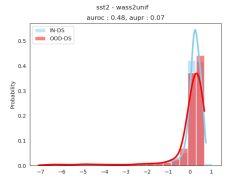
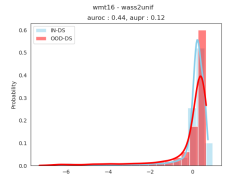
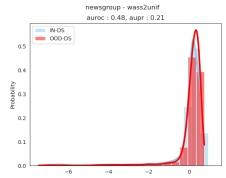
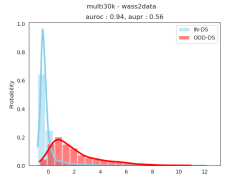
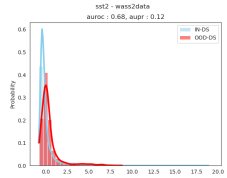
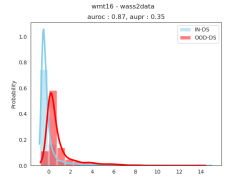
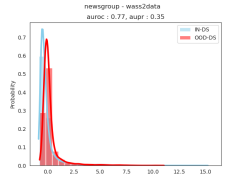
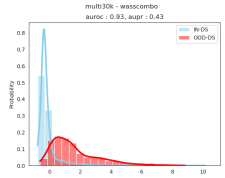
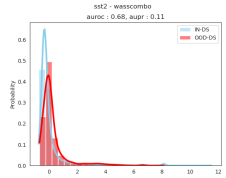
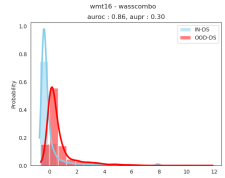
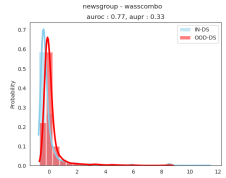
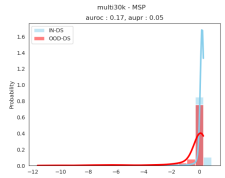
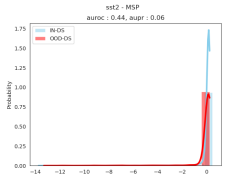
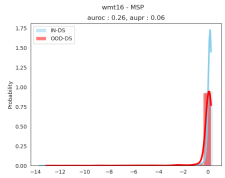
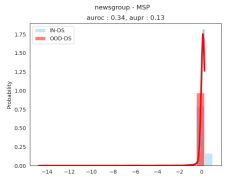
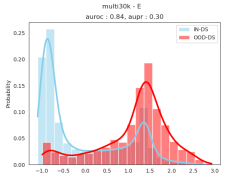
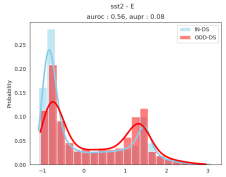
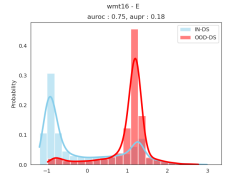
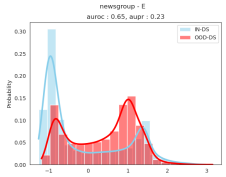
Method \TestSet	multi30k	sst2	wmt16	newsgroup
TRUSTED				
Mahalanobis				
LiLO				
Wass2Unif				
Wass2Data				
WassCombo				
MSP				
E				

Table 3: Benchmark of methods

D LiLO BenchMark

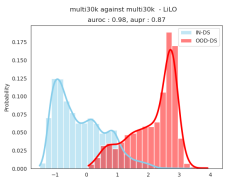
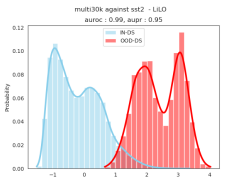
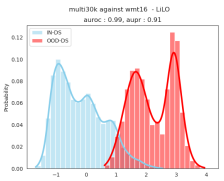
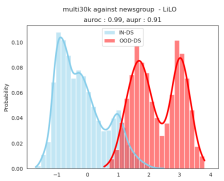
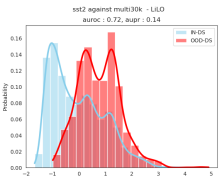
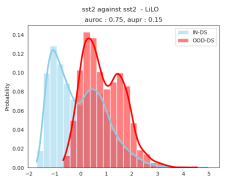
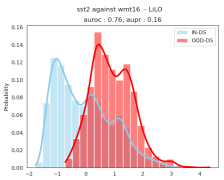
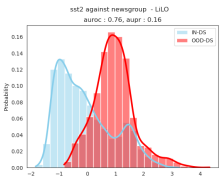
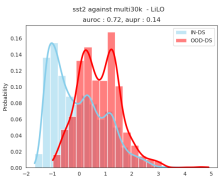
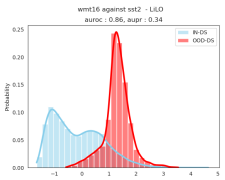
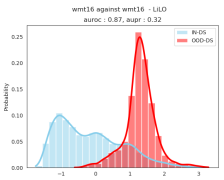
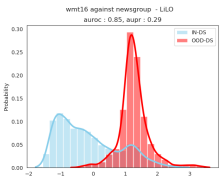
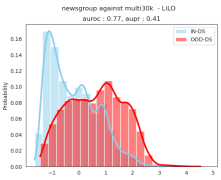
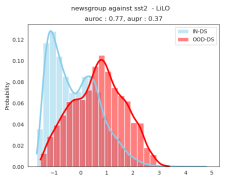
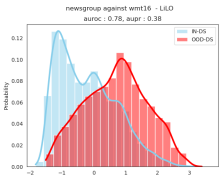
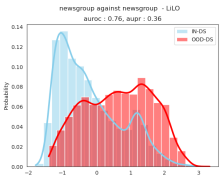
Testset \Trainset	multi30k	sst2	wmt16	newsgroup
multi30k				
sst2				
wmt16				
newsgroup				

Table 4: Benchmark of LiLO,
IN-DS = imdb, random sampling = 10%

E Exponential aggregation Err values on BERT for IMDB vs SST-2

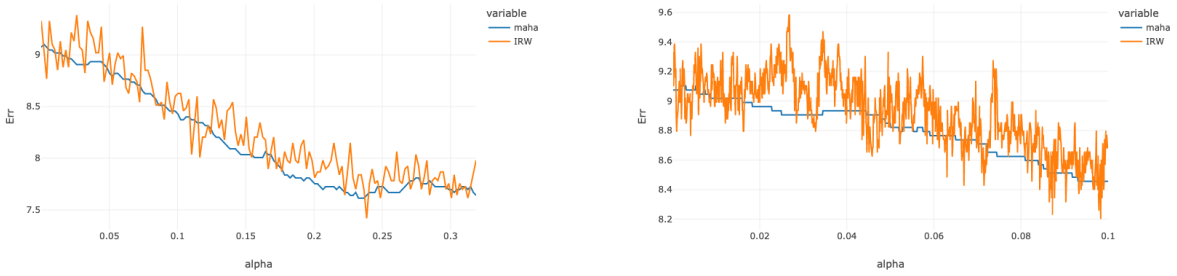


Figure 5: Err value for $\alpha \in [0.001, 0.32]$ (left) and $\alpha \in [0.001, 0.01]$ (right)