Lightweight Out-of-Distribution Detection for Patent Classification in Non-Stationary Environments

Ekaterina Kotliarova[1], Sebastian Björkqvist[1]

[1]IPRally Technologies Oy, Helsinki, Finland

Abstract

Categorizing patents into different classes is an essential step in processes such as monitoring competitors, managing patent portfolios, and landscaping existing inventions. In practical applications, classifiers are often trained on limited data and then applied to out-of-distribution documents, i.e., samples that are quite different from what the classifier was trained on. This may result in incorrect and nonsensical classification results. In this work, we explore lightweight methods for detecting such out-of-distribution (OOD) samples before classification. We show that a simple nearest neighbor-based approach is highly reliable for OOD sample detection in general, with the downside of having to store the embeddings of the training set to perform inference. We also introduce a method based on probability density functions (PDF) and show that when combined with a custom thresholding strategy, it effectively retains in-distribution samples and filters out anomalies, while requiring the storage of only the mean and covariance matrix of the training data.

Keywords

classification, distribution shift, anomaly detection, patents, document embeddings, patent search

1. Introduction

Classifying patent documents plays a central role in various industrial and legal processes. In practical deployments, classifiers often operate on limited and evolving data, and may be applied to domains different from those they were trained on. These conditions lead to distribution shifts between training and test data, and therefore to the appearance of out-of-distribution (OOD) inputs, i.e., documents that differ significantly from those seen by the model during training.

Our previous work addressed several of these challenges by utilizing search-based embeddings and semi-supervised learning to improve classification with limited data [1, 2]. As a continuation, this paper focuses on detecting OOD-inputs before classification to prevent unreliable predictions in mismatched domains.

To this end, we evaluate several unsupervised OOD-detection methods operating in the embedding space. In particular, we suggest a lightweight approach that: **(i) has high in-distribution (ID) retention:** the method retains almost all relevant documents; **(ii) is easy-to-use:** the method requires a minimal computation, *i.e.,* only the empirical mean and covariance matrix of the training data are computed, while test samples are scored via multivariate Gaussian probability density function (PDF); **(iii) and is OOD-agnostic:** *i.e.,* the detection threshold is calibrated using only provided by user training ID-data.

2. Literature survey

Modern machine learning models are often developed under the assumption that training and test data are drawn from the same underlying distribution [3], but this rarely holds in practice. In open-world settings, models frequently encounter out-of-distribution (OOD) inputs –samples that differ significantly from the training data [4].

6th Workshop on Patent Text Mining and Semantic Technologies (PatentSemTech) 2025

ekaterina@iprally.com (E. Kotliarova); sebastian@iprally.com (S. Björkqvist)

0000-0002-5491-7741 (E. Kotliarova); 0009-0006-9039-8623 (S. Björkqvist)

As discussed in [5], distributional shifts in tasks based on textual data representations can generally be categorized into two types: **(i) semantic shift**, where the OOD samples belong to entirely new categories and should not be mapped to any existing class; **(ii) non-semantic shift**, where OOD samples differ in a domain or style but share the same class semantics as the ID samples.

Our task falls under the semantic shift scenario, where the OOD documents may come from previously unseen patent categories and must not be forced into known classes. As mentioned in [5], we can thus utilize the following taxonomy of OOD-detection methods to detect semantic shift: **(i)** the OOD samples available for the training; **(ii)** the OOD samples are unavailable, but the ID labels available; **(iii)** both the OOD data and the ID label unavailable.

In our setting, while the training set is labeled for further classification, the labels do not contribute to the OOD-detection step, which is inherently unsupervised. This leads us to conclude that the OOD-detection task in our case falls back to the third option, which is a well-known classic anomaly detection problem [6].

3. Methodology

Although various strategies can be applied to perform OOD-detection, in this work, we focus on evaluating lightweight, unsupervised OOD-detection methods in the embedding space. Specifically, we compare a PDF-based likelihood approach with common baselines such as k-NN, Local Outlier Factor, and Isolation Forest (see the discussion in Section 3.1). As thresholding heavily impacts final performance, we also examine different threshold selection strategies (Section 3.2).

3.1. Baseline and Proposed Methods

As discussed in Section 2, our task falls under unsupervised OOD-detection, which requires no OOD samples or class labels during training. We prioritize lightweight methods with continuous scores, allowing us to control the strictness of OOD-detection by adjusting the decision threshold, while keeping training costs low. The goal of OOD-detection is to suppress unreliable classification outputs for inputs that deviate significantly from the training data. Thus, detected OOD-samples can be withheld from further classification or flagged for manual review. We selected the following methods for our evaluations:

1. Distance-based methods, such as k-nearest neighbors (k-NN), which compute the average distance of a test point to its k closest training embeddings [7]. The idea behind nearest-neighbor methods is that ID (in-distribution) data are more likely to be closer to its neighbors than OOD data. After computing the scores, a threshold is applied (the threshold selection is covered in Section 3.2).
2. Density-based methods, such as Local Outlier Factor (LOF) and Isolation Forest (IF), which estimate how isolated a test sample is compared to the ID data[8, 9]. Both methods compute continuous anomaly scores and require a threshold.
3. Likelihood-based models, which estimate the probability of a sample under a distribution fitted to the training data. In our case, we adopt a custom Probability Density Function (PDF)-based approach. By computing the mean and covariance of the in-distribution (ID) training set, we evaluate the likelihood of each test sample under this distribution. The method is described in detail in Algorithm 1.

We use implementations provided by the *scikit-learn* library for the k-Nearest Neighbors, Local Outlier Factor, and Isolation Forest algorithms [10]. Meanwhile, our PDF-based approach works as presented in Algorithm 1.

3.2. Threshold computation

Threshold selection plays a crucial role in OOD-detection, as it directly influences the final result; therefore, it should be considered an important part of the overall approach. A common practice is to set the threshold to achieve a high true positive rate (TPR) on in-distribution data and then report the

Algorithm 1 OOD-Detection via PDF

Input: Training dataset D_{in}, test sample x^*, threshold λ
Training Stage:
1. Compute mean μ_{in} and covariance matrix Σ_{in} of train dataset D_{in}.
Inference Stage:
1. Compute the OOD-score of the test sample x^* by computing the probability density (*i.e.,* $p(x^*)$) under the multivariate Gaussian distribution P_{in} defined by the training data's mean μ_{in} and covariance Σ_{in}.
2. Compare computed OOD-score to the threshold λ; if $p(x^*) \geq \lambda$ the sample is considered in-distribution (ID), otherwise it is flagged as out-of-distribution (OOD).
Output: Binary decision whether x^* is from the same distribution P_{in} as training data D_{in} (ID) or not (OOD).

corresponding false positive rate (FPR) on OOD data. While many works report FPR@95%TPR meaning the threshold is set so that 95% TPR on the validation set is achieved [5, 7], in our case, retaining the relevant documents is the priority, so we instead use a 99% TPR score threshold.

Additionally, for the PDF-based method (Algorithm 1), we utilize a custom threshold selection algorithm. The method aims to avoid using unstable low-probability outliers as the threshold, while also not enforcing arbitrary strictness, such as discarding a fixed percentage of data. To achieve this, we smooth the normalized likelihoods and identify approximate inflection points. The lowest such point serves as a cutoff, and all values below this point are considered to be outliers. The fact that the cutoff is not the global minimum is crucial, since inflection point detection is approximate due to the smoothing used, and the global minimum is often unusable due to zero-likelihood artifacts. See Fig. 1 for the intuition behind this approach. The final threshold is set as the minimum of the outlier-cleaned set. The full procedure is detailed in Algorithm 2.

Algorithm 2 Custom Threshold Selection

Input: Validation set D_{val}, mean μ_{in}, covariance matrix Σ_{in} of train dataset D_{in}
Threshold Selection:
1. Compute likelihoods of D_{val} using mean μ_{in} and covariance matrix Σ_{in} as presented in Inference Stage of Algorithm 1, resulting in a set L_{val}.
2. Normalize values in L_{val} to the range $[0, 1]$.
3. Apply Gaussian smoothing with bandwidth σ to normalized likelihood set L_{val}. The parameter σ controls how much the curve would be smoothed. We use $\sigma = 2.0$, selected empirically.
4. Compute second derivative of smoothed scores.
5. Identify inflection points: locations where second derivative changes sign. Note that the inflection points are approximate due to normalization and computational artifacts.
6. Identify the minimum among the inflection points. Use this value as a cutoff: remove all likelihood values in L_{val} that are equal to or lower than this point, yielding a cleaned set $L_{\text{val-clean}}$.
7. Set threshold $\lambda = \min\{L_{\text{val-clean}}\}$.
Output: Threshold λ.

3.3. Graph-based embeddings trained for patent search

We use embeddings generated as described in [13, 14], where each patent document is first converted into a graph that represents the key features of the invention and their relationships. The resulting graphs are significantly smaller than the original documents, which enables efficient processing of large documents while still preserving relevant information required for prior art searches. The graph is then embedded into a vector space using a graph neural network (GNN) trained to perform prior art searches using patent examiner citation data. Using citation data enables the model to recognize semantically

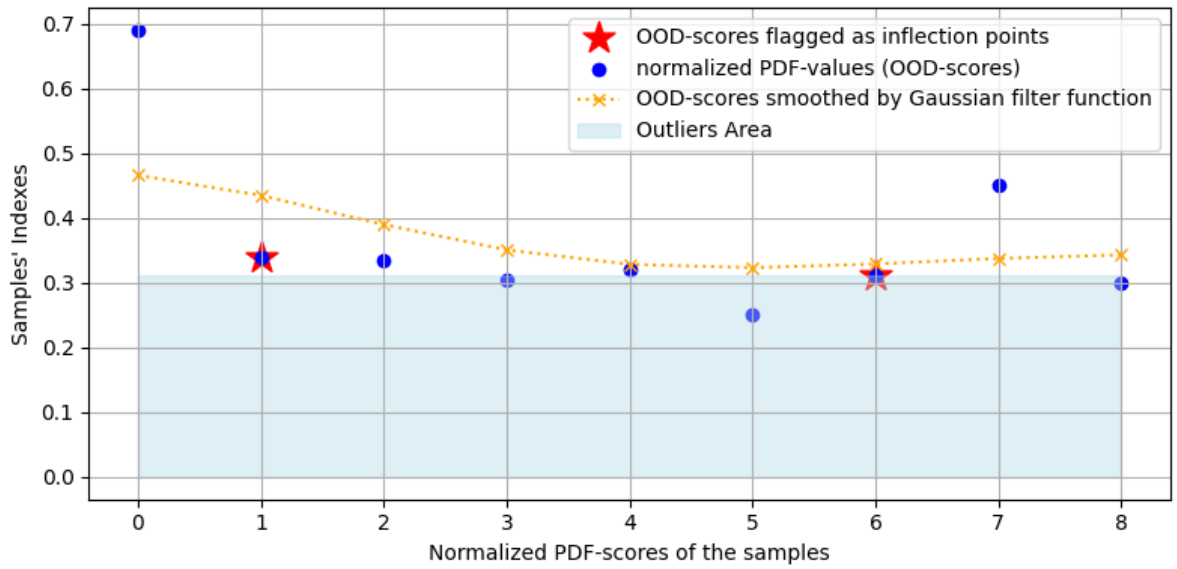

Figure 1: Illustration of the idea behind the inflection point search. The method identifies two approximate inflection points (red stars), which are sufficient to isolate low-probability outliers in the validation set. While all true inflection points are detected, the approach separates unstable scores near zero.

Dataset	Train	Validation	Test
Qubit [11]	896	224	280
Cannabinoid [12]	941	235	293
Mechanical eng.	2868	717	896
Chemical	810	202	253

Table 1
Number of samples in the datasets used for training and evaluation. Only one document per family is kept in each data set to avoid overrepresentation of large patent families.

similar inventions despite differences in terminology, placing them close together in the embedding space. The resulting embeddings may used as input to a lightweight classification model, as shown in [1, 2].

3.4. Datasets

Four datasets were chosen for this study: two public and two proprietary. The public datasets are the Qubit [11] and the Cannabinoid patent datasets [12]. The proprietary ones originate from distinct domains: one from the mechanical engineering patent domain and the other from the chemistry field. Only one document per family is kept in each data set to avoid over-representation of large patent families (refer to Table 1 for the dataset sizes).

For the purpose of evaluation, we simulate an OOD-detection setup by selecting one dataset (e.g., Qubit) to serve as the in-distribution (ID) set and treating samples from the remaining datasets as out-of-distribution (OOD). All four datasets originate from different domains, which reflects realistic domain shift scenarios in patent classification.

3.5. Experiment setup and evaluation

Each model is trained using a training set extracted from the complete dataset. The models take document embeddings as input and generate OOD-scores for each test sample as output. Validation sets are fixed for every dataset. For the experiments on the subsets of data we partition the training set by randomly sampling p percent of the data points with p ranging from 5 to 100.

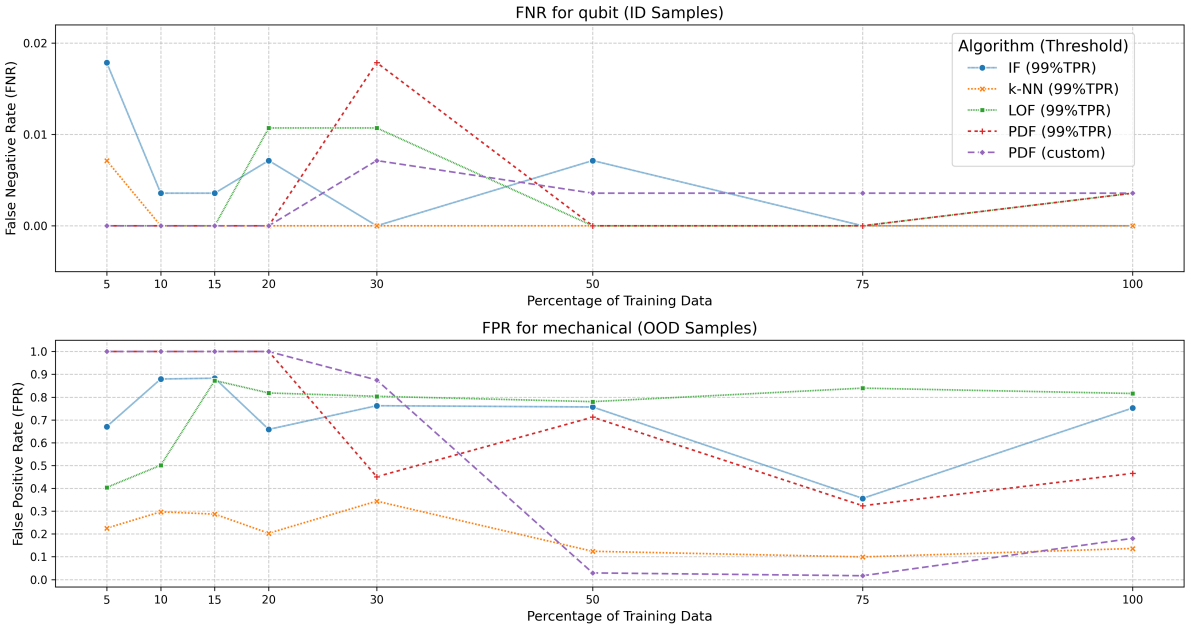

Figure 2: False Negative Rate (FNR) for ID samples (top) and False Positive Rate (FPR) for OOD samples (bottom) on the Qubit training dataset and Mechanical test dataset. While the k-NN algorithm shows robust results for all subsets of data, ID and OOD cases, $\text{PDF}_{\text{custom}}$ achieves similar performance for the bigger train data subsets.

	ID (Qubit)			OOD (Mechanical)			OOD (Chemical)			OOD (Cannabinoid)		
%	$\text{PDF}_{\text{custom}}$ FNR	PDF FNR	k-NN FNR	$\text{PDF}_{\text{custom}}$ FPR	PDF FPR	k-NN FPR	$\text{PDF}_{\text{custom}}$ FPR	PDF FPR	k-NN FPR	$\text{PDF}_{\text{custom}}$ FPR	PDF FPR	k-NN FPR
5	**0.0**	**0.0**	0.007	1.0	1.0	**0.226**	1.0	1.0	**0.407**	1.0	1.0	**0.362**
15	**0.0**	**0.0**	**0.0**	1.0	1.0	**0.287**	1.0	1.0	**0.162**	1.0	1.0	**0.167**
30	0.007	0.018	**0.0**	0.875	0.459	**0.344**	0.261	0.261	**0.107**	0.993	0.335	**0.201**
50	0.004	**0.0**	**0.0**	**0.029**	0.712	0.124	**0.06**	0.099	0.08	0.205	0.28	**0.106**
100	0.004	0.004	**0.0**	0.181	0.465	**0.137**	**0.032**	0.134	0.047	**0.038**	0.12	0.058

Table 2
Evaluation results with Qubit as the training dataset (in-distribution, ID) and Mechanical, Chemical, and Cannabinoid as out-of-distribution (OOD) test sets. The percentage symbol (%) indicates the portion of the Qubit dataset used during training. Lower FNR (for ID) and FPR (for OOD) are preferred. The PDF and k-NN scores were computed using a standard TPR99% threshold (Section 3.2), while $\text{PDF}_{\text{custom}}$ uses a threshold computed via Algorithm 2.

When training on a subset of the data, we repeat random sampling and model training n times to reduce the noise caused by the data splits, where n ranges from 6 for the largest subsets to 100 for the smallest. The metrics from multiple sampling iterations for the same percentage are then averaged.

To make a binary decision based on the OOD-score, we compute the threshold based on the ID validation set so that 99% TPR is achieved, as discussed in Section 3.2. Additionally for the PDF-based approach, a custom threshold is utilized based on the Algorithm 2.

Evaluations for all subsets of data were conducted using a separate holdout test set, independent of the training data. We report FPR@99%TPR–the false positive rate at the threshold that achieves a 99% true positive rate. For the ID test samples FNR@99%TPR is shown.

4. Results and discussions

The results of all methods are shown in Fig. 2, with the Qubit dataset chosen as the in-distribution data set. The false negative rate (FNR) hovers around 1% for all methods, which is to be expected since we selected the threshold to achieve 99% TPR. The k-NN algorithm has a low false positive rate (FPR) on

all training data set sizes, while our PDF-based method combined with the custom threshold selection achieves similar or lower FPR as k-NN when at least 50% of the training data set is used. The other algorithms have significantly higher FPR.

Table 2 presents how the k-NN and PDF algorithms perform using other OOD data sets. The k-NN algorithm is the most stable, performing well even with small amounts of training data, while the PDF method combined with the custom threshold selection performs well when at least 50% of the training data is used. The results also demonstrate the usefulness of the custom threshold selection algorithm. If the threshold for the PDF method is set to achieve 99% TPR then the FPR is significantly higher with large data sets.

Future work could explore various thresholding strategies for the k-NN method and explore modifications that reduce the need to store the entire training set to calculate distances between test samples and k-nearest-neighbors. Perhaps, the method could be adapted to operate using the mean vector or a set of representative cluster centers instead.

5. Conclusions

In this work we analyzed algorithms for detecting OOD-samples in classification. We demonstrated that using nearest neighbors achieves the best trade-off between detecting OOD-samples and keeping ID-samples, especially with small training sets. We also introduced a PDF-based method and showed that it, when combined with a custom threshold selection algorithm, works well with large training sets while avoiding the need to store the entire training set to perform inference.

Declaration of Generative AI

During the preparation of this work, the authors used Grammarly in order to: Grammar and spelling check. And further ChatGPT-4o in order to: Paraphrase and reword as well as to improve writing style. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] J. Lagus, E. Kotliarova, S. Björkqvist, Patent classification on search-optimized graph-based representations, in: Proceedings of the 4th Workshop on Patent Text Mining and Semantic Technologies, PatentSemTech'23, 2023, pp. 33–38. URL: https://ceur-ws.org/Vol-3604/paper2.pdf.

[2] E. Kotliarova, S. Björkqvist, Semi-supervised learning methods for patent classification using search-optimized graph-based representations, in: Proceedings of the 5th Workshop on Patent Text Mining and Semantic Technologies, PatentSemTech'24, 2024, pp. 18–24. URL: https://ceur-ws.org/Vol-3775/paper4.pdf.

[3] V. Vapnik, Principles of risk minimization for learning theory, in: Proceedings of the 5th International Conference on Neural Information Processing Systems, NIPS'91, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1991, p. 831–838.

[4] A. Bendale, T. Boult, Towards open world recognition, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1893–1902. doi:10.1109/CVPR.2015.7298799.

[5] H. Lang, Y. Zheng, Y. Li, J. Sun, F. Huang, Y. Li, A survey on out-of-distribution detection in NLP, Transactions on machine learning research (2023). URL: https://par.nsf.gov/servlets/purl/10526541.

[6] J. Gu, Y. Ming, Y. Zhou, J. Kuen, V. Morariu, H. Zhao, R. Zhang, N. Barmpalios, A. Liu, Y. Li, T. Sun, A. Nenkova, A critical analysis of document out-of-distribution detection, in: H. Bouamor, J. Pino, K. Bali (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2023, Association for Computational Linguistics, Singapore, 2023, pp. 4973–4999. URL: https://aclanthology.org/2023.findings-emnlp.332/. doi:10.18653/v1/2023.findings-emnlp.332.

[7] Y. Sun, Y. Ming, X. Zhu, Y. Li, Out-of-distribution detection with deep nearest neighbors, in: K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, S. Sabato (Eds.), Proceedings of the 39th International Conference on Machine Learning, volume 162 of *Proceedings of Machine Learning Research*, PMLR, 2022, pp. 20827–20840. URL: https://proceedings.mlr.press/v162/sun22d.html.

[8] M. M. Breunig, H.-P. Kriegel, R. T. Ng, J. Sander, Lof: identifying density-based local outliers, in: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, SIGMOD '00, Association for Computing Machinery, New York, NY, USA, 2000, p. 93–104. URL: https://doi.org/10.1145/342009.335388. doi:10.1145/342009.335388.

[9] F. T. Liu, K. M. Ting, Z.-H. Zhou, Isolation forest, in: 2008 Eighth IEEE International Conference on Data Mining, 2008, pp. 413–422. doi:10.1109/ICDM.2008.17.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[11] S. Harris, A. Trippe, D. Challis, N. Swycher, Construction and evaluation of gold standards for patent classification—a case study on quantum computing, World Patent Information 61 (2020) 101961.

[12] S. Harris, Gold standard for the evaluation of machine classification of patent data, 2019. URL: https://github.com/swh/classification-gold-standard/tree/master.

[13] S. Björkqvist, J. Kallio, Building a graph-based patent search engine, in: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 3300–3304. URL: https://doi.org/10.1145/3539618.3591842. doi:10.1145/3539618.3591842.

[14] K. Daniell, I. Buzhinsky, S. Björkqvist, Efficient patent searching using graph transformers, in: Proceedings of the 6th Workshop on Patent Text Mining and Semantic Technologies, PatentSemTech'25, 2025. To appear.