Open-World Classification with Bayesian Gaussian Mixture Models

Extended Abstract

Justin Clarke
University of Massachusetts
Amherst, United States
jclarke@cs.umass.edu

Przemyslaw Grabowicz
University College Dublin
Dublin, Ireland
przemek.grabowicz@ucd.ie

David Jensen
University of Massachusetts
Amherst, United States
djensen@cs.umass.edu

ABSTRACT

Methods for solving classification tasks often assume a data generating process with stable structure that remains fixed during both training and inference. However, autonomous agents deployed in real-world environments often perform classification in situations where the data generating process is dynamic and the ontology of classes is only partially known. Such tasks are known as *open-world classification* (OWC). We present *open-world mixture modeling* (OMM), a framework for OWC using Bayesian Gaussian mixture models. With only slight modifications to the standard Bayesian variational inference algorithm, we are able to detect and model novel classes as they appear in a data stream, while maintaining and updating the classes learned during training. Empirical evaluations reveal that the method reliably detects novel classes with performance similar to a supervised classifier trained on labeled samples of the novel classes.

KEYWORDS

Open-World Learning; Multi-Class Classification; Continual Learning

ACM Reference Format:
Justin Clarke, Przemyslaw Grabowicz, and David Jensen. 2025. Open-World Classification with Bayesian Gaussian Mixture Models: Extended Abstract. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 3 pages.

1 INTRODUCTION

Consider the problem of an autonomous robot operating in a remote wilderness environment with the task of estimating the populations of large animal species in a given area. Each time the robot captures an image of an animal, it attempts to accurately classify the species to ensure accurate population estimates. A classifier could be trained in advance using images of all the species expected to be in the area, but what if the robot encounters an unfamiliar species? Ideally, the classifier would identify the animal as novel and not a member of any of the species seen during training. Additionally, the ideal classifier would correctly identify any future animals that belong to the novel species. Finally, the ideal classifier would improve its model of the novel species as more examples are observed. Such an ideal classifier requires capabilities beyond those of existing machine learning methods.

In this work we introduce *open-world mixture modeling* (OMM), a framework for open-world classification (§2). We develop an agent based on a Bayesian Gaussian mixture model with a flexible number of components that grows with the complexity of the data. Empirical evaluation on synthetic and real-world datasets shows that the method is effective at identifying and modeling new components (§3). Despite the simplicity of OMM, we are not aware of any prior work that uses mixture models to address OWC.

2 AGENT DESIGN

Constructing an open-world mixture model (OMM) begins by training a Bayesian Gaussian mixture model in a supervised manner using data that is representative of the known classes in the environment.

When a batch of data arrives, the agent must determine if it contains any samples from new classes that were not seen during training. Many existing approaches to this task define boundaries around known classes and set a threshold on how far a new sample must be from an existing class boundary to be considered novel. Although these methods are somewhat successful [4], the OMM takes a different approach by modeling a candidate novel class and then using the information gathered to identify novel samples.

The model cannot be certain which samples from the batch, if any, belong to the novel class, so it initializes the candidate novel component via empirical Bayes, using the mean and covariance of the batch data inversely weighted by the likelihood of each sample under the current model. This has the effect of up-weighting samples that are least likely under the existing model, and therefore most likely to be samples from a new class.

To refine its estimate of the parameters of a potential novel class, the model uses Bayesian variational EM [1], including the potential new component. The strength of the priors on the existing components' parameters are usually much higher than for the potential new component, so the parameter values of non-novel components are more stable during EM.

Then, OMM compares the effective number samples assigned to the candidate component k, $N_k = \sum_{n=1}^{N} p(z_k = 1|\mathbf{x}_n)$, to a detection threshold hyperparameter to determine if there is sufficient evidence to update the model. This is an important difference between OMM and many other existing approaches. OMM does not need to define a strict border for each class or a minimum distance threshold to qualify as a novel sample. Instead, the results of EM identify which samples should be considered novel and then OMM can make

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

an application specific decision if the evidence is sufficient to update the model. The ability to avoid additions of invalid novel classes distinguishes OMM from typical non-parametric Bayesian clustering approaches [2], which assume that the number of components always increases with the number of samples.

3 EXPERIMENTAL RESULTS

3.1 Novelty Detection

To analyze the agent's ability to detect samples from novel classes, we simulate a scenario where the agent has been trained on an initial set of known classes and then observes a single batch of new data that includes a varying number of N_{novel} samples from a novel class.

For each value of N_{novel} we generate 10 unique class configurations by iterating through the classes in the MNIST dataset, where each class takes a turn serving as the novel class and three random classes are selected to be used as the non-novel classes [3]. The initial model is trained using 2000 samples from each of the known classes, where the $28x28$ images are normalized, flattened into vectors, and reduced to 100 dimensions using PCA. The agent then observes a single batch of new data with an additional 500 samples from each of the known classes, and a varying number of samples from the novel class in the range $[50, 1000]$. For each component configuration we repeat the sampling process five times, which varies the specific novel samples being detected, but not their number. The novelty detection threshold hyperparameter, which is the minimum number of novel samples to be considered a valid detection, is set to 75.

Figure 1 (Top) shows the binary detection rate as a function of the number of novel samples in the batch (green line). We also show that the rate of false detections is low over the same component configurations when no novel samples are present in the batch (red line), which demonstrates the agent's ability to avoid false detections and refrain from introducing unnecessary model components. To evaluate whether the agent accurately identifies which samples should be modeled by the new model component, we report the accuracy of the instantiated novel component (orange), as well as the accuracy (purple) and F1-score (blue) for the complete model using a held out test set. This reflects the accuracy of the model immediately after the detection process, before any additional parameter updates.

3.2 Novelty Adaptation

To evaluate the agent's ability to adapt to novel classes in a data stream, we create an idealized DGP consisting of a set of Gaussian components with random parameters uniformly sampled from the range $[-100, 100]$, and a marginal probability distribution representing the prior probability of a sample being associated with each component. Data is generated sequentially by first randomly selecting a component according to the marginal distribution, and then sampling from that component using the current parameters. With each batch, there is a probability p_{new} that a new component will be added to the DGP, and the marginal component probabilities re-normalized.

The agent begins with two non-novel classes consisting of 400 samples each. We then generate new data in batches, each with $N_{batch} = 1000$ samples. For each sample we generate its x value. For each batch with probability $p_{new} = 0.25$ we generate parameters for a new Gaussian component and add it to the DGP. We set the marginal probability of all components to be equal, and normalize them accordingly. This ensures a reasonable chance of detection on the first attempt while still making the problem challenging. Finally, at each step we evaluate the OMM's classification accuracy by generating a test set consisting of 200 samples for each class in the DGP, regardless of whether they were detected by the agent.

Figure 1 (bottom) shows the agent's accuracy when classifying samples on the current test set after each batch is processed. The vertical lines indicate when a new component was added to the DGP. We find that the OMM performs nearly as well as a Gaussian mixture model trained in a supervised setting, where labels of all "novel" classes were provided during training.

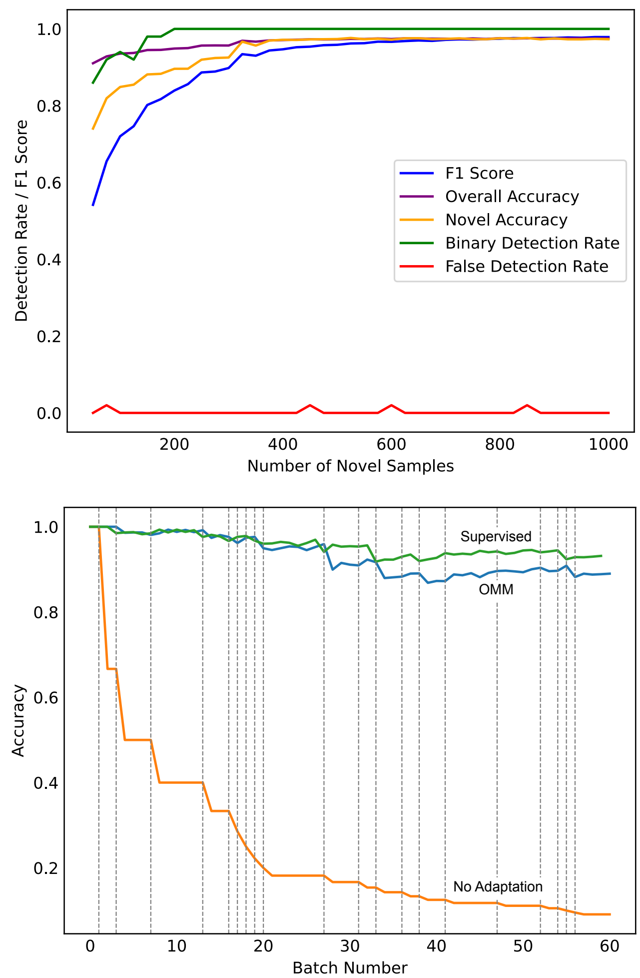

Figure 1: (Top) Novelty detection performance on the MNIST dataset as a function of the number of novel samples in the batch. (Bottom) Accuracy after each batch of synthetic data on a held out test set containing samples from all classes in the stream.

REFERENCES

[1] Christopher M. Bishop. 2007. *Pattern Recognition and Machine Learning (Information Science and Statistics)* (1 ed.). Springer.

[2] David M. Blei and Michael I. Jordan. 2006. Variational inference for Dirichlet process mixtures. *Bayesian Analysis* 1, 1 (2006), 121 – 143. https://doi.org/10.1214/06-BA104

[3] Yann LeCun, Corinna Cortes, and CJ Burges. 2010. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* 2 (2010).

[4] Ethan M. Rudd, Lalit P. Jain, Walter J. Scheirer, and Terrance E. Boult. 2015. The Extreme Value Machine. *CoRR* abs/1506.06112 (2015). arXiv:1506.06112 http://arxiv.org/abs/1506.06112