LDA for Ranking

Introduction and Motivation

We know that certain patches of instances in the feature space are anomalies. The detectors which try to detect these patches are not perfect and usually have mismatches in their assumptions about the structure of the data. An example of this is (say) using a Gaussian Mixture Model with 5 clusters to model data that actually contains 8 clusters. As a consequence, each detector correctly identifies only a subset of the anomalies with high probability.

Another common aspect of model driven anomaly detection is that when some anomalies are close to each other in the feature space, then if a detector identifies one anomaly in a particular region, it will very likely identify other anomalies in its proximity. Therefore, groups of anomalies are either identified together with high probability by a detector or they are missed out together.

The above two conditions suggest that if we see the outputs of multiple detectors, we should see groups of data instances that co-occur frequently at similar (anomaly) ranks – especially at the top. And if this is the case, we should expect a few underlying dominant themes of anomaly ranking. Technically, each such theme defines a distribution of anomaly ranks over the data instances (Figure 1). Assuming such dominant themes exist, a consensus algorithm should first figure out the latent distributions from the (noisy) rankings across all detectors. Next, it should generate a composite ranking such that if an instance is ranked high under any of the distributions then it should get a higher final rank. In addition, the consensus should also account for the popularity of the distributions among detectors.

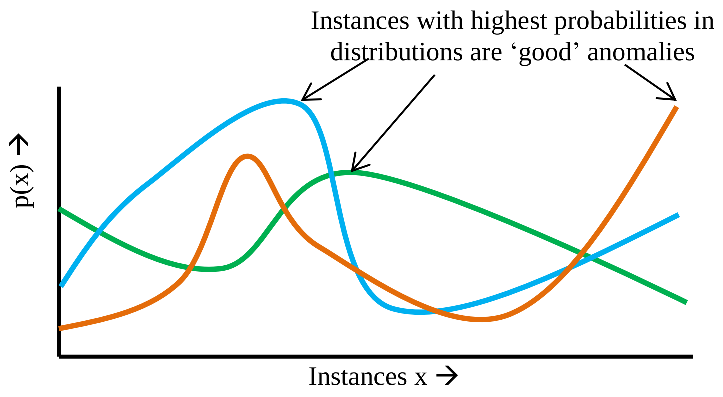
Instances with highest probabilities in distributions are 'good' anomalies
p(x) →
Instances x →

Figure 1: Each curve represents a latent rank distribution. The instances are not ordered in any particular manner. *p(x)* is representative of the probability with which an instance appears near the top in anomaly ranking. The most promising anomalies are those at the peaks of the distributions.

LDA topic models are very well suited to the task of identifying the latent distributions. For this, we need to translate the ranking outputs of detectors to a document model with count data. We explain one approach below.

Translating the Detector Outputs to Documents

Assume that our ensemble contains *D* detectors each of which provides a complete ranking over *N* data instances. We treat one detector's ranking output as one *document*. The instances are treated as the *vocabulary*. So, there are *D* documents and *N* words. To use a multinomial model like LDA which requires count data, one option is to transform the ranks (or scores) in some way such that they correspond to relative word frequencies in the documents. The current implementation uses a vote assignment strategy which is somewhat ad hoc and might be refined:

1. We first fix a fraction *f* such that only the highest-ranked *f* fraction of data instances will be assigned non-zero votes for each detector. In other words, only *f* fraction of words within a document will have a non-zero count. By default, we set this fraction to 0.5 (i.e., half the instances are assigned non-zero votes by a detector).

2. Among the top *f* fraction instances, the top ranked instance gets the highest number of votes v_{max} (we fix this to 50 by default.) v_{max} should be set higher (say, between 100 and 500) if the number of instances is very large as in ADAMS.
3. The vote count decays at an exponential rate and falls to 1 for the instance that ranked last among the top *f* fraction of instances.

We now have the rank outputs as bag-of-word documents. The fraction *f* determines the sparsity of the words in documents. See Table 1 for an illustration.

Model	Inst_1	Inst_2	Inst_3	Inst_4	Inst_5	Inst_6	Inst_7	…	Inst_N
EGMM	30	45	10	4	3	0	12	…	27
GMM	29	41	8	0	6	5	10	…	32
…								…	…
RDE	12	0	32	22	26	0	21	…	5

Table 1: This is an illustration of how detector outputs are translated to documents. Each row is a document corresponding to one detector output. The columns are words and correspond to data instances. The value in each cell represents the number of votes assigned to an instance.

Identify Latent Distributions with LDA

Under LDA, the latent distributions correspond to topics. We use a simple LDA model to compute these topics. Assume that there are *T* latent topics and each detector contains a mixture of one or more topics. The generative process for ranking is:

1. Select $\boldsymbol{\varphi}_t \sim Dirichlet(\boldsymbol{\alpha})$
2. For each Detector (document)
    a. Select $\boldsymbol{\theta_d} \sim Dirichlet(\boldsymbol{\beta})$
    b. For each Instance (word) in the Detector
        i. Select topic $t \sim Mult(\boldsymbol{\theta_d})$
        ii. Select $votes \sim Mult(\boldsymbol{\varphi}_t)$

The consensus algorithm will not know beforehand how many latent topics might be present. It might be possible to infer this somehow, but for now the advice is to be conservative and assume the smallest reasonable number. For example, if there are only seven detectors in the ensemble, then a reasonable number of topics might be just 2.

Consensus Ranking

After training, LDA returns the parameters $\boldsymbol{\Theta} = \{\boldsymbol{\theta_d}\}_{d=1}^{D}$ and $\boldsymbol{\Phi} = \{\boldsymbol{\varphi_t}\}_{t=1}^{T}$. Individually, each $\boldsymbol{\varphi_t}$ assigns an anomaly probability to each word/instance within the topic *t*. For a composite score, we weight the anomaly scores by topic popularity: $\boldsymbol{\theta}^p = \sum_{d=1}^{D} \boldsymbol{\theta_d}$. The final score is: $\boldsymbol{S} = \boldsymbol{\Phi}\boldsymbol{\theta}^p$. Note that $\boldsymbol{\Phi}$ is an $N \times T$ matrix and $\boldsymbol{\theta}^p$ is a $T \times 1$ matrix. Therefore we get $\boldsymbol{S}$, an $N \times 1$ matrix. Within S, the higher values correspond to higher likelihood of being anomaly.