Universiteit
Leiden
The Netherlands

Unsupervised representation learning for time series anomaly detection and beyond

Ferreira Correia; L.

Citation

Unsupervised representation learning for time series anomaly detection and beyond. (2026, September 8). *Unsupervised representation learning for time series anomaly detection and beyond*. Retrieved from https://hdl.handle.net/1887/4309435

Version:	Publisher's Version
License:	Licence agreement concerning inclusion of doctoral thesis in the Institutional Repository of the University of Leiden
Downloaded from:	https://hdl.handle.net/1887/4309435

Note: To cite this publication please use the final published version (if applicable).

Chapter 6

Interactive Threshold Search Using Active Learning

6.1 Introduction

As concluded in Chapter 5.4, there is a need for a methodology which does not assume labelled data is available for free, like is often the case in the literature. To meet this need, active learning can be employed to intelligently choose samples to query labels from a domain expert. The labelled data can then be used to choose a threshold which maximises the F_1 score, similar to the method used to obtain the hypothetical best threshold τ_{best} in Chapter 5.2. The process of choosing samples to be queried is referred to as a query strategy (QS) and is explained in more detail in Chapter 2.6. This chapter is structured as follows. First, Chapter 6.2 provides an overview of other ways active learning is applied to anomaly detection problems. Then, in Chapter 6.3, a novel QS is proposed, named the dissimilarity-based query strategy (DQS), which queries samples most dissimilar to each other, in theory maximising the diversity of labelled sequences. To quantify and compare the detection performance uplifts brought about by DQS, experiments are outlined where it is benchmarked against three other QSs not only using different query budget sizes, but also different mislabelling probabilities. The latter aspect is relevant to the real-world, as even domain experts may make mistakes, though unfortunately the impact of mislabelling has not yet been investigated in literature. In Chapter 6.4 the results of the experiments are then presented and analysed, and finally, in Chapter 6.5, insights gained are once again

recapped and future work is highlighted.

Mathematically, the problem in this chapter can be described as follows. Consider an anomaly detection problem involving some sort of dynamic process running for several days that yields a number of multivariate time series that are sequential w.r.t. each other but are not necessarily temporally contiguous, i.e. a discrete-sequence anomaly detection problem as described in Chapter 2.2. During the dynamic process, each recorded time series $\mathcal{S}$ is added to the candidate set $\mathcal{C}^{\text{ts}} = \{\mathcal{S}_1, ..., \mathcal{S}_n, ..., \mathcal{S}_N\}$ containing all recorded sequences up to the present, where N denotes the number of sequences recorded up to the present and $\mathcal{S}_n \in \mathbb{R}^{T_n \times d_{\mathcal{D}}}$, where T_n denotes the number of time steps in $\mathcal{S}_n$, and $d_{\mathcal{D}}$ the dimensionality of the data set. Now, consider an anomaly detector $\mathcal{M}$ that has been used to check the data for anomalous behaviour and yields a univariate anomaly score $\mathbf{s}_n = \mathcal{M}(\mathcal{S}_n)$, where $\mathbf{s}_n \in \mathbb{R}^{T_n}$. The anomaly score counterpart to the candidate set $\mathcal{C}^{\text{ts}}$ is denoted as the candidate *score* set $\mathcal{C}^{\text{as}}$ and is structured similarly, such that $\mathcal{C}^{\text{as}} = \{\mathbf{s}_1, ..., \mathbf{s}_n, ..., \mathbf{s}_N\}$.

In the context of active learning with time series data, a sequence is referred to as a sample, where a subset of sequences of $\mathcal{C}^{\text{ts}}$ is chosen to be sent to the oracle, i.e. the domain expert. Once the query budget B is used up, i.e. B time series are chosen, they are moved to a set called the query set $\mathcal{Q}^{\text{ts}}$ which contains all time series selected for querying up to the present. The anomaly score counterpart to the query set $\mathcal{Q}^{\text{ts}}$ is denoted as the query score set $\mathcal{Q}^{\text{as}}$ and is structured similarly. Once the query score set is established, it can be used to improve the anomaly detection performance using the feedback strategy (FS).

6.2 Related Work

The wide majority of model-based approaches found in the literature use some sort of anomaly score, which indicates how anomalous each time step within a test subset sequence is. To classify time steps in the anomaly score as nominal or anomalous, a threshold needs to be set, however, many publications in unsupervised anomaly detection circumvent the issue by using labelled data, technically making the respective approaches supervised. The labelled data is either used to set a threshold which satisfies a condition, like achieving the maximum possible F_1 score or some other parametric threshold setting method is proposed, like the peaks over threshold method [87], which has found wide application throughout the literature [77, 88–90]. Hundman et al. [32] introduce the non-parametric dynamic threshold, but unfortunately, despite *non-parametric* being in the name, it is not truly non-parametric, as the threshold

can only be set by setting a hyperparameter **z** which is experimentally set, and therefore requires labelled data. In addition to that, this method cannot be applied to discrete-sequence anomaly detection, as it assumes a predicted anomalous time steps or ground-truth anomalous time steps within a given test sequence. When neither is given, like in the case of a correctly identified nominal time series, the threshold becomes undefined.

Generally, active learning is rarely used to adjust thresholds in model-based time series anomaly detection approaches. Literature dealing with further training the anomaly detection model exists, but such methods are not generically applicable, since they modify loss functions to penalise wrong classifications [105–108] or utilise an architecture specific to active learning which also requires labelled data from the beginning [109].

Lundström et al. [110], on the other hand, propose a threshold setting method for continuous-sequence problems. First, it initialises a threshold based on a variation of the "non-parametric" dynamic threshold [32], then, anomalous sub-sequences are clustered according to four pre-defined attributes of temporally adjacent sub-sequences. According to Lundström et al. [110] the mathematical definition of *temporally adjacent* implies that two anomalous sub-sequences can be temporally adjacent if they occur after one another, regardless of whether there is another sub-sequence between them, though in this work their explanation is interpreted as referring to two sub-sequences following one another *directly*, i.e. there is no other anomalous sub-sequence between them. The first attribute, A_1 denotes the temporal distance between the last time step of a predicted anomalous sub-sequence and the first time step of the following predicted anomalous sub-sequence. The second and third attributes, A_2 and A_3, denote the absolute difference between the respective maximums of two temporally adjacent predicted anomalous sub-sequences. The difference between the second and third attributes is that the former looks for the maximum value in each channel of the input time series, whereas the latter looks for the maximum value in each channel of the anomaly score. The fourth attribute A_4 is a binary vector, which assumes a 1 if both maximum anomaly scores in each channel of two temporally adjacent subsequences exceed the current threshold. As is evident, all but the first attribute are obtained feature-wise, i.e. $A_2, A_3, A_4 \in \mathbb{R}^{d_{\mathcal{D}}}$. Following the calculation of attributes, Lundström et al. [110] aggregate them into several logical outputs, which assume a binary value depending on whether the respective attribute exceeds the set threshold for that attribute. The logical outputs are then combined to decide whether two temporally adjacent predicted anomalous sub-sequences should be clustered together. Interestingly,

this threshold setting method requires pre-defined thresholds to work, and therefore it needs to be parametrised. Additionally, it cannot be applied to discrete-sequence problems since, to obtain attribute A_1, predicted anomalous sub-sequences need to be within the same sequence, which may not always be the case.

There is a clear need for methods which allow for the integration of active learning into the threshold choice for discrete-sequence anomaly detection problems. Additionally, all the above-mentioned literature assumes perfect labelling, however, as pointed out by Settles [111], labels from human experts are not always reliable as some samples are difficult to label and humans can be distracted or fatigued over time. Therefore, this work investigates the impact of different rates of mislabelling on the anomaly detection performance, which has not been undertaken previously in the literature.

6.3 Proposed Methodology

6.3.1 Dissimilarity-based Querying Strategy

Generally, query strategies work by letting an oracle label B times each round, i.e. the number of samples allowed to be queried in a given round q. While it is possible to choose B sequences randomly every round, there is little control over the diversity of the queried sequences. In this work, the diversity of sequences is maximised by actively choosing samples most dissimilar to the previously queried ones, by inspecting the dynamic time warping (DTW) distance between the samples in question, i.e. the anomaly scores of different time series, where a high distance is interpreted as poor alignment of features within the anomaly scores and, therefore, a high level of dissimilarity. For a more detailed explanation of DTW, please refer to Chapter 2.7. Unlike other metrics, like the mean-squared-error, dynamic time-warping distance does not require two sequences to be the same length, which makes it ideal for an application with variable-length sequences. Compared to univariate time series, calculating the dissimilarity between multivariate time series in the same way is possible, however, it is much more computationally intensive.

The exact procedure for DQS is as follows and is also depicted as pseudocode in Algorithm 6. For the first query in the first round, i.e. $b = q = 1$ and $|\mathcal{Q}^{\text{as}}| = 0$, DQS is initialised by choosing a random anomaly score with index r from the candidate score set $\mathcal{C}^{\text{as}}$, as shown in line 3 in Algorithm 6. It is then moved to the query score set $\mathcal{Q}^{\text{as}}$, such that $|\mathcal{Q}^{\text{as}}| = 1$, as shown in line 16. Once the querying is initialised, for the next query, i.e. $b = 2$, the anomaly score in $\mathcal{C}^{\text{as}}$ most similar to any anomaly score

Algorithm 6 DQS at an arbitrary round q

Require: Candidate score set $\mathcal{C}^{\text{as}}$, Query score set $\mathcal{Q}^{\text{as}}$, Budget B

1: **for** $b = 1 \rightarrow B$ **do**
2: **if** $|\mathcal{Q}^{\text{as}}| = 0$ **then**
3: $r \sim \mathcal{U}(1, |\mathcal{C}^{\text{as}}|)$ ▷ Random index for initialisation
4: **else**
5: **for** $n = 1 \rightarrow |\mathcal{C}^{\text{as}}|$ **do**
6: **for** $m = 1 \rightarrow |\mathcal{Q}^{\text{as}}|$ **do**
7: $\mathbf{D}[n, m] = DTW(\mathcal{C}^{\text{as}}[n], \mathcal{Q}^{\text{as}}[m])$
8: **end for**
9: **end for**
10: $o = \arg\min(\mathbf{D})$ ▷ Index most similar to any sample in $\mathcal{Q}^{\text{as}}$
11: **for** $n = 1 \rightarrow |\mathcal{C}^{\text{as}}|$ **do**
12: $\mathbf{E}[n] = DTW(\mathcal{C}^{\text{as}}[n], \mathcal{C}^{\text{as}}[o])$
13: **end for**
14: $r = \arg\max(\mathbf{E})$ ▷ Index most dissimilar to $\mathcal{C}^{\text{as}}[o]$
15: **end if**
16: $\mathcal{Q}^{\text{as}} \leftarrow \mathcal{C}^{\text{as}}[r]$ ▷ Move $\mathcal{C}^{\text{as}}[r]$ to $\mathcal{Q}^{\text{as}}$
17: **end for**
18: **return** Query Score Set $\mathcal{Q}^{\text{as}}$

within $\mathcal{Q}^{\text{as}}$ is tagged; its index is denoted as o, as shown in lines 5 to 10. Following that, the anomaly score in $\mathcal{C}^{\text{as}}$ most dissimilar to the tagged $\mathcal{C}^{\text{as}}[o]$, denoted by index r, is moved to $\mathcal{Q}^{\text{as}}$, such that $|\mathcal{Q}^{\text{as}}| = 2$, as shown in lines 11 to 14. This is repeated until B sequences are moved to $\mathcal{Q}^{\text{as}}$, such that, at the end of round 1, $|\mathcal{Q}^{\text{as}}| = B$. For round 2, no initialisation is required, and hence the procedure is the same as round 1 for $b > 1$. Therefore, after round 2, B anomaly scores are moved to the query score set $|\mathcal{Q}^{\text{as}}| = 2 \cdot B$. The following rounds after that follow the same pattern. Clearly, the query set and the candidate set both grow with every round, complicating computation as time goes on.

Once a round is over and B new samples are added to the query score set $\mathcal{Q}^{\text{as}}$, the threshold τ is grid-searched to maximise the F_1 score based on the labels provided by the oracle, though future work could investigate the feasibility of more efficient search methods.

This method is generically applicable to discrete-sequence problems that rely on a threshold as a decision boundary between nominal and anomalous sequences.

Obviously, a QS can only be employed in cases where a domain expert is available. Cases where this does not apply will benefit from more sophisticated ways to choose a truly unsupervised threshold.

6.3.2 Experimental Setup

Competing Approaches

As mentioned in Chapter 6.1, DQS is benchmarked against three other QS. The first, named the random-based query strategy (RQS), represents a scenario where, rather than intelligently querying an oracle, the samples to be labelled are chosen at random, as shown in Algorithm 7.

Algorithm 7 RQS at an arbitrary round q

Require: Candidate Score Set $\mathcal{C}^{\text{as}}$, Query Score Set $\mathcal{Q}^{\text{as}}$

for $b = 1 \rightarrow B$ **do**
  $r \sim \mathcal{U}(1, |\mathcal{C}^{\text{as}}|)$ ▷ Random index is sampled
  $\mathcal{Q}^{\text{as}} \leftarrow \mathcal{C}^{\text{as}}[r]$
end for
return Query Score Set $\mathcal{Q}^{\text{as}}$

The second, named the top-based query strategy (TQS) [105–107, 109, 112, 113], represents a scenario where the B samples with the highest anomaly score are queried, as shown in Algorithm 8. Some literature [27, 114, 115] also refers to this QS as uncertainty-based, as they interpret a high anomaly score as high uncertainty.

Algorithm 8 TQS at an arbitrary round q

Require: Candidate Score Set $\mathcal{C}^{\text{as}}$, Query Score Set $\mathcal{Q}^{\text{as}}$

for $b = 1 \rightarrow B$ **do**
  $r = \arg\max(\mathcal{C}^{\text{as}})$ ▷ Index of maximum anomaly score
  $\mathcal{Q}^{\text{as}} \leftarrow \mathcal{C}^{\text{as}}[r]$
end for
return Query Score Set $\mathcal{Q}^{\text{as}}$

The third, named the uncertainty-based query strategy (UQS), represents a scenario where the B samples closest to the threshold are queried, as shown in Algorithm 9. This strategy needs to be initialised with a threshold, however, in literature, this is not always defined [107, 112, 113] or it relies on labelled data [109].

Benchmarking Considerations

This work performs all experiments on the powertrain anomaly time series benchmark (PATH) data set, a discrete-sequence data set that provides a large number of diverse of multivariate time series and non-trivial anomalies. The dataset represents the dynamic behaviour of an automotive powertrain at different states, which is generated using

Algorithm 9 UQS at an arbitrary round q

Require: Candidate Score Set $\mathcal{C}^{\text{as}}$, Query Score Set $\mathcal{Q}^{\text{as}}$
if $|\mathcal{Q}^{\text{as}}| = 0$ **then**
  $\bar{\mathcal{C}}_n^{\text{as}} = \frac{1}{T_n} \sum_{t=1}^{T_n} \mathbf{s}_n$
  $\tau = \frac{1}{N} \sum_{n=1}^{N} (\bar{\mathcal{C}}_n^{\text{as}})$
end if
for $b = 1 \rightarrow B$ **do**
  $r = \arg\min(\text{abs}(\mathcal{C}^{\text{as}} - \tau))$ ▷ Index of nearest anomaly score to τ
  $\mathcal{Q}^{\text{as}} \leftarrow \mathcal{C}^{\text{as}}[r]$
end for
return Query Score Set $\mathcal{Q}^{\text{as}}$

state-of-the-art simulation tools to exhibit real-world-like complexity, as described in Chapter 4.1. Other publicly available datasets exist, though none are of discrete-sequence nature and this size and diversity, as shown in Chapter 3.1. To adapt the PATH data set for active learning, the unlabelled data subset is split such that the time series within it represent a day's worth of measurements, two days worth of measurements, etc. until the entire subset is depleted, which results in 34 increasingly larger splits. Therefore, for the day 1 split, the sum of the durations of each sequence within the split is roughly equal to 24 hours, for the day 2 split the sum is roughly equal to 48 hours, and so forth. This mimics the gradual collection of data with time, like in the real world. To maintain reasonable computational complexity, it is not feasible to perform active learning every day, therefore a querying round of B samples is done after day 1, day 7, day 14, day 21, and day 28. On each of the days considered, a new model is also trained, such that the new threshold obtained using active learning is applied to the newly trained model. Like in Chapter 5.2, the validation subset size is set as 20 % of the unlabelled subset.

To provide context to the results obtained for the different QS, two baselines will also be provided to the results. The first represents a scenario where the unsupervised threshold τ_{us} is used after each model training, whereas the second represents a hypothetical scenario where the best threshold τ_{best} is known. It is obtained by grid-searching through the test subset and choosing the threshold yielding the highest F_1 score. Neither of the scenarios involve active learning, i.e. no samples are sent to the oracle for labelling.

Like in Chapter 5.3, the online evaluation metrics are obtained for each fold, though unlike in that chapter, the modelling process is not the focus. The query and feedback strategies occur *after* model training and highly depend on the seed used due to

the random operations in the initialisation of the query strategies and mislabelling mechanism; therefore, the results for each fold are collected obtained using seeds 1 to 3, after which the average for each evaluation metric is provided.

The active learning-based approaches tested in this work always use the most recent model available and assume that the oracle is queried once every day at the end of the day. For each split, labels for queried sequences in earlier splits are also considered.

Environment

Temporal VAE (TeVAE) is used as the underlying model and is trained as specified in Chapter 5.2, except with varying training subset sizes. The framework used for model training is TensorFlow 2.15.1 and TensorFlow Probability 0.23 on Python 3.10 on a workstation running Ubuntu 22.04.5 LTS, equipped with two Nvidia RTX A6000 GPUs. Further information on library versions used can be found in the *requirements.txt* file in the GitHub repository under `github.com/lcs-crr/Thesis`.

6.4 Results

6.4.1 Baselines without Active Learning

First, a selection of baseline results is presented in Table 6.1. The table shows the results obtained using the unsupervised threshold τ_{us}, i.e. the threshold obtained by taking the maximum value anomaly score observed in the validation subset, and the results obtained hypothetical best threshold τ_{best}, i.e. the threshold that maximises the F_1 score. The latter is referred to as *hypothetical* since it is obtained using the test subset, which in the real-world is not available, making this threshold unobservable outside this experimental setting.

Table 6.1: F_1 scores for each of the splits using the unsupervised threshold τ_{us} and the hypothetical best threshold τ_{best}.

Baseline	1 Day	7 Days	14 Days	21 Days	28 Days
Unsupervised	0.16 ± 0.03	0.16 ± 0.01	0.12 ± 0.10	0.06 ± 0.04	0.04 ± 0.02
Best	0.20 ± 0.05	0.38 ± 0.09	0.45 ± 0.10	0.58 ± 0.13	0.68 ± 0.10

As expected, the results using the hypothetical best threshold τ_{best} are higher across the board. There is a downward trend in the F_1 score obtained with the unsupervised threshold τ_{us} as time goes on and the validation subset grows. This is because, with a

larger validation subset and therefore a higher anomaly count within it, the probability of a very high anomaly score grows, leading to a higher-than-ideal threshold setting. *The goal of any QS is to outperform the unsupervised threshold τ_{us}, while coming as close to the hypothetical best as possible.*

6.4.2 Impact of Query Budgets

A large query budget B is associated with a higher labelling cost, and hence it is desirable to use a QS that maximises the F_1 score improvement with a limited number of queries. To investigate the impact the query budget has on the anomaly detection results, three different budget sizes are used: $B = 1$, $B = 5$, and $B = 10$. In this experiment, it is assumed that all queries are correctly labelled.

The results for different query strategies (random-based, top-based, uncertainty-based, dissimilarity-based) and different query budgets ($B = 1$, $B = 5$, $B = 10$) are presented in Table 6.2 and, to further aid interpretation, as a function of time in Figure 6.1.

Table 6.2: F_1 scores for each of the splits for different query budgets using different query strategies and the DQS. The results obtained using the unsupervised threshold τ_{us} and the hypothetical best threshold τ_{best} are also presented for reference. The best results for the respective split at a given budget size B is shown in **bold**. The values shown consist of the mean ± standard deviation over the three folds, each on the three different seeds. A plot of the F_1 score as a function of time is shown in Figure 6.1.

	B	1 Day	7 Days	14 Days	21 Days	28 Days
τ_{us}	-	0.16 ± 0.03	0.16 ± 0.01	0.12 ± 0.10	0.06 ± 0.04	0.04 ± 0.02
τ_{best}	-	0.20 ± 0.05	0.38 ± 0.09	0.45 ± 0.10	0.58 ± 0.13	0.68 ± 0.10
RQS	1	$\mathbf{0.10 \pm 0.01}$	0.10 ± 0.01	0.10 ± 0.01	0.10 ± 0.01	0.16 ± 0.16
TQS	1	$\mathbf{0.10 \pm 0.01}$	0.19 ± 0.13	0.14 ± 0.05	0.21 ± 0.14	0.24 ± 0.17
UQS	1	$\mathbf{0.10 \pm 0.01}$	0.10 ± 0.01	0.10 ± 0.01	0.13 ± 0.04	0.14 ± 0.06
DQS	1	$\mathbf{0.10 \pm 0.01}$	$\mathbf{0.28 \pm 0.11}$	$\mathbf{0.31 \pm 0.14}$	$\mathbf{0.39 \pm 0.18}$	$\mathbf{0.53 \pm 0.19}$
RQS	5	0.11 ± 0.02	0.16 ± 0.06	0.29 ± 0.18	0.39 ± 0.23	0.50 ± 0.25
TQS	5	0.14 ± 0.06	$\mathbf{0.28 \pm 0.06}$	0.29 ± 0.02	0.30 ± 0.06	0.26 ± 0.12
UQS	5	0.14 ± 0.06	0.17 ± 0.11	0.17 ± 0.11	0.20 ± 0.15	0.20 ± 0.14
DQS	5	$\mathbf{0.15 \pm 0.05}$	0.27 ± 0.07	$\mathbf{0.31 \pm 0.15}$	$\mathbf{0.43 \pm 0.19}$	$\mathbf{0.61 \pm 0.13}$
RQS	10	0.12 ± 0.03	0.18 ± 0.06	0.27 ± 0.17	0.44 ± 0.22	$\mathbf{0.60 \pm 0.15}$
TQS	10	$\mathbf{0.16 \pm 0.05}$	$\mathbf{0.28 \pm 0.07}$	0.31 ± 0.12	0.44 ± 0.06	0.51 ± 0.14
UQS	10	$\mathbf{0.16 \pm 0.05}$	0.18 ± 0.07	0.22 ± 0.14	0.27 ± 0.12	0.35 ± 0.18
DQS	10	$\mathbf{0.16 \pm 0.05}$	$\mathbf{0.28 \pm 0.06}$	$\mathbf{0.39 \pm 0.10}$	$\mathbf{0.50 \pm 0.12}$	0.46 ± 0.23

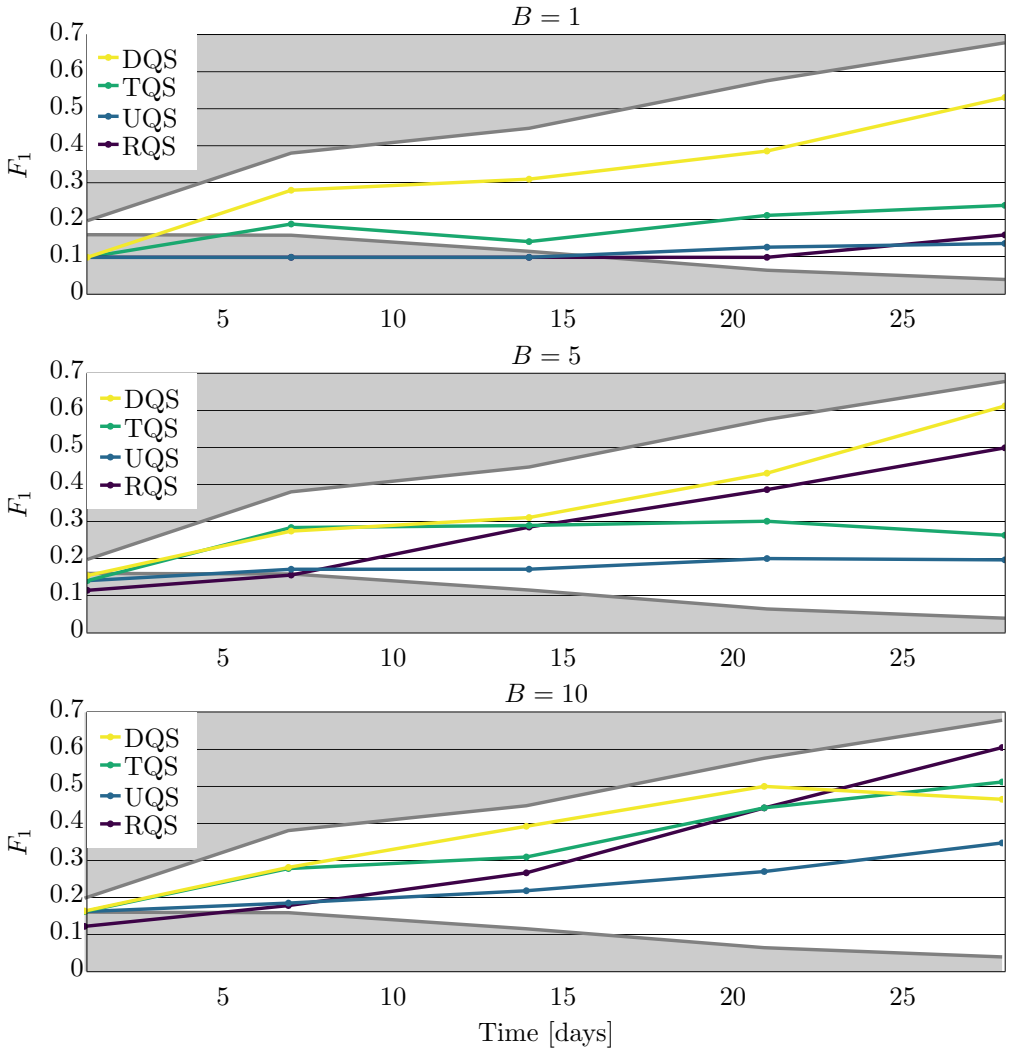

Figure 6.1: F_1 scores for different query budgets at a fixed mislabelling probability of $p_m = 0$ plotted as a function of time. The results in tabular form are also shown in Table 6.2. The grey upper bound represents the results using the hypothetical best threshold τ_{best}, and the grey lower bound represents the results using the unsupervised threshold τ_{us}.

The first observation that can be made from Table 6.2 and Figure 6.1 is that, at the smallest split of 1 day for all budget sizes, all query strategies perform worse or, at most, comparably to the scenario using the unsupervised threshold τ_{us}. This can be attributed to the poor model quality at the small training subset size, indicated by the low hypothetical best results. *As the training subset grows, the difference between*

the two baselines grows, which is where active learning can have the most impact. At small budget sizes, i.e. at $B < 10$, the DQS outperforms the other query strategies across the board. At $B = 10$, the DQS outperforms or is at most matched by the other query strategies except for the last split of 28 days, where the RQS performs the best. Lastly, at most budget sizes and splits, the UQS falls short of the other query strategies.

6.4.3 Impact of Mislabelling Probability

While it is assumed the oracle in this work is a domain expert, they can still make mistakes. In literature, the effect of mislabelling has not been investigated as far the author is aware, therefore, for a fixed budget size of $B = 10$ queries, a mislabelling probability of $p_m = 0.1$, $p_m = 0.2$, $p_m = 0.3$ is experimented with. This is done by sampling from a Bernoulli distribution parameterised with the above-mentioned mislabelling probabilities and flipping ground-truth labels accordingly. The choice of the Bernoulli distribution is based on several reasons. For one, unlike a Binomial distribution, which is a function of the number of trials, the Bernoulli distribution is of single-trial nature. Additionally, the Bernoulli distribution is discrete and binary, sampling it yields a boolean, corresponding to mislabelled or correctly labelled.

The results for different query strategies (random-based, top-based, uncertainty-based, dissimilarity-based) and different mislabelling probabilities ($p_m = 0.1$, $p_m = 0.2$, $p_m = 0.3$) are presented in Table 6.3 and, to further aid interpretation, as a function of time in Figure 6.2.

Unsurprisingly, the above-made observations regarding the low F_1 scores at the smallest split, can also be made when investigating the impact of mislabelling on detection performance. Aside from the results at the smallest split, all query strategies perform comparably at $p_m = 0.1$, with DQS performing slightly better. This holds mostly true for $p_m = 0.2$ and $p_m = 0.3$ as there is no clear winner for all split sizes. What is clear, is that at the two higher mislabelling probabilities, the RQS performs worst. Overall, the results scatter more between query strategies as the mislabelling probability increases. Interestingly, some results improve despite higher mislabelling probabilities. This can happen when the mislabelled samples do not change the position of the threshold, for example, when an equal number of samples are mislabelled on either side of the threshold.

Table 6.3: F_1 scores for each of the splits for a query budget of $B = 10$ and different mislabelling probabilities using different query strategies and the DQS. The results obtained using the unsupervised threshold τ_{us} and the hypothetical best threshold τ_{best} are also presented for reference. The best results for the respective split at a given mislabelling probability p_m is shown in **bold**. The values shown consist of the mean ± standard deviation over the three folds, each on the three different seeds. A plot of the F_1 score as a function of time is shown in Figure 6.2.

	p_m	1 Day	7 Days	14 Days	21 Days	28 Days
τ_{us}	-	0.16 ± 0.03	0.16 ± 0.01	0.12 ± 0.10	0.06 ± 0.04	0.04 ± 0.02
τ_{best}	-	0.20 ± 0.05	0.38 ± 0.09	0.45 ± 0.10	0.58 ± 0.13	0.68 ± 0.10
RQS	0	0.12 ± 0.03	0.18 ± 0.06	0.27 ± 0.17	0.44 ± 0.22	$\mathbf{0.60 \pm 0.15}$
TQS	0	$\mathbf{0.16 \pm 0.05}$	$\mathbf{0.28 \pm 0.07}$	0.31 ± 0.12	0.44 ± 0.06	0.51 ± 0.14
UQS	0	$\mathbf{0.16 \pm 0.05}$	0.18 ± 0.07	0.22 ± 0.14	0.27 ± 0.12	0.35 ± 0.18
DQS	0	$\mathbf{0.16 \pm 0.05}$	$\mathbf{0.28 \pm 0.06}$	$\mathbf{0.39 \pm 0.10}$	$\mathbf{0.50 \pm 0.12}$	0.46 ± 0.23
RQS	0.1	0.11 ± 0.02	0.21 ± 0.06	0.25 ± 0.15	0.38 ± 0.17	0.53 ± 0.21
TQS	0.1	$\mathbf{0.16 \pm 0.05}$	0.27 ± 0.07	0.32 ± 0.06	0.44 ± 0.19	0.50 ± 0.13
UQS	0.1	$\mathbf{0.16 \pm 0.05}$	0.19 ± 0.09	0.30 ± 0.12	0.43 ± 0.22	0.43 ± 0.11
DQS	0.1	0.13 ± 0.05	$\mathbf{0.29 \pm 0.06}$	$\mathbf{0.36 \pm 0.13}$	$\mathbf{0.50 \pm 0.15}$	$\mathbf{0.57 \pm 0.17}$
RQS	0.2	0.12 ± 0.02	0.17 ± 0.06	0.20 ± 0.14	0.20 ± 0.10	0.43 ± 0.17
TQS	0.2	$\mathbf{0.16 \pm 0.05}$	$\mathbf{0.33 \pm 0.08}$	$\mathbf{0.33 \pm 0.06}$	0.43 ± 0.05	0.51 ± 0.15
UQS	0.2	$\mathbf{0.16 \pm 0.05}$	0.23 ± 0.11	0.29 ± 0.12	0.44 ± 0.24	0.46 ± 0.16
DQS	0.2	0.11 ± 0.04	0.23 ± 0.10	$\mathbf{0.33 \pm 0.13}$	$\mathbf{0.46 \pm 0.17}$	$\mathbf{0.54 \pm 0.17}$
RQS	0.3	0.11 ± 0.02	0.15 ± 0.06	0.20 ± 0.16	0.15 ± 0.05	0.21 ± 0.12
TQS	0.3	$\mathbf{0.14 \pm 0.06}$	$\mathbf{0.35 \pm 0.06}$	0.28 ± 0.12	$\mathbf{0.52 \pm 0.15}$	0.47 ± 0.14
UQS	0.3	0.14 ± 0.06	0.26 ± 0.12	$\mathbf{0.31 \pm 0.12}$	0.37 ± 0.26	0.40 ± 0.15
DQS	0.3	0.11 ± 0.04	0.23 ± 0.11	0.30 ± 0.17	0.33 ± 0.21	$\mathbf{0.55 \pm 0.15}$

6.5 Conclusions

The results show that, there is no QS that outperforms all others in every category. While DQS performs best in small-budget cases and remains competitive at higher ones, TQS demonstrates higher robustness when faced with mislabelling. In the real world, it therefore depends on the expertise of the oracle and how many samples they are willing to label. In any case, except for very small splits, all querying strategies outperform the unsupervised threshold τ_{us} despite the presence of mislabelling, and hence, based on the experiments undertaken on the PATH dataset, it is advisable to use an active learning-based threshold if the possibility to query an oracle exists.

Future work should focus on standardising the benchmarking procedure of active learning methods in time series anomaly detection and should include investigation

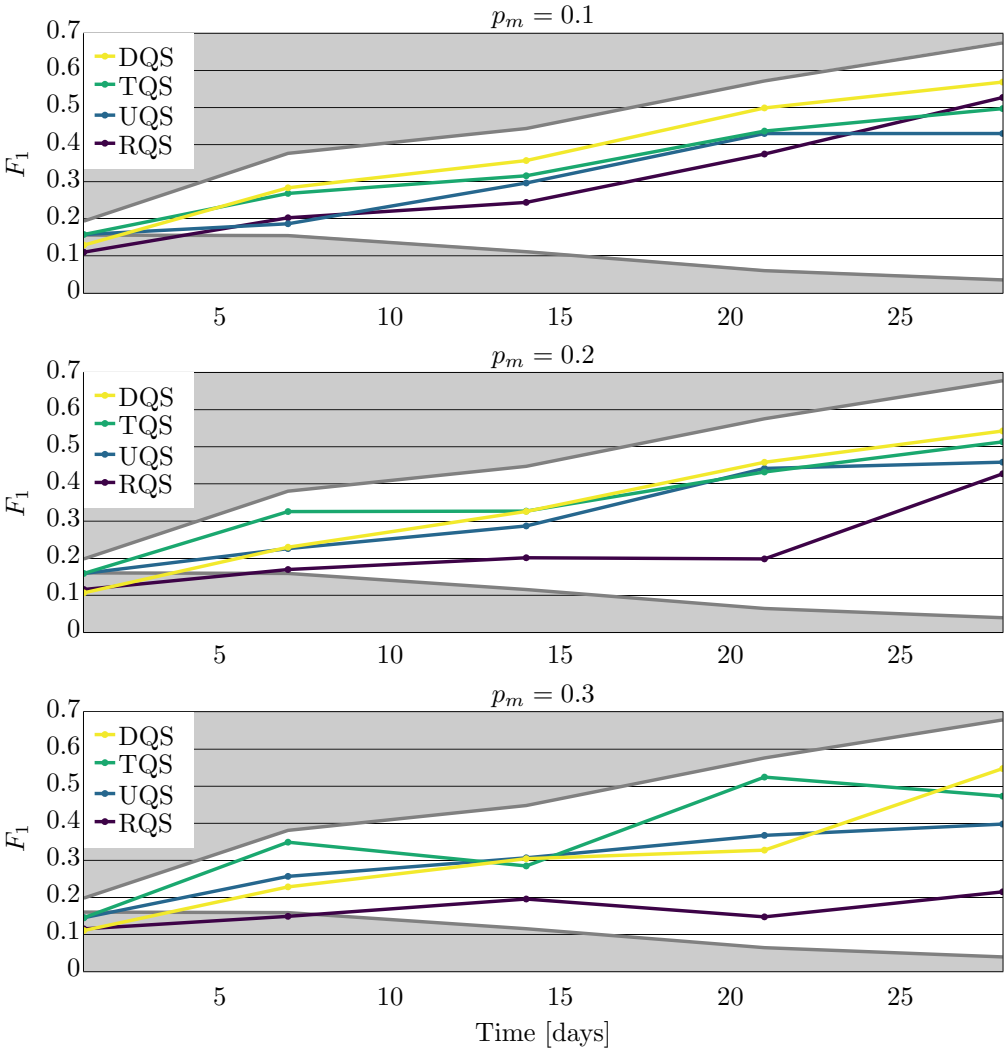

Figure 6.2: F_1 scores for different mislabelling probabilities at a fixed query budget of $B = 10$ plotted as a function of time. The results in tabular form are also shown in Table 6.3. The grey upper bound represents the results using the hypothetical best threshold τ_{best}, and the grey lower bound represents the results using the unsupervised threshold τ_{us}.

on the impact of mislabelling, proposed in this work. Furthermore, creating a QS that not only performs well with small query budgets but is also robust in the case of mislabelling should be further researched. Perhaps this can be achieved by measuring dissimilarity differently, either by using of a different metric besides DTW or by comparing other aspects, like the reconstructions in the case of autoencoders.