Neural Assessment of Non-Destructive Assay for Material Accountancy

Randall Gladen
Pacific Northwest National Laboratory[*]
randall.gladen@pnnl.gov

Tom Grimes
Pacific Northwest National Laboratory[*]
thomas.grimes@pnnl.gov

Ben Wilson
Pacific Northwest National Laboratory[*]
benjamin.wilson@pnnl.gov

Jack Dermigny
Pacific Northwest National Laboratory[*]
jack.dermigny@pnnl.gov

Nathan Shoman
Sandia National Laboratories
nshoman@sandia.gov

Benjamin B. Cipiti
Sandia National Laboratories
bbcipit@sandia.gov

Abstract

The IAEA spends significant effort verifying the non-diversion of nuclear materials at bulk handling facilities such as reprocessing plants and uranium enrichment plants. This verification often requires that the IAEA collect and analyze dozens of small samples taken from the process to directly confirm the concentration of uranium and/or plutonium in the process materials. The collection of these samples is often resource intensive for both the IAEA and facility operator and the subsequent IAEA analysis is quite expensive. Given the increasing amounts of nuclear material under IAEA safeguards and their fixed budget, significant improvements to IAEA resource could be achieved if these samples could be reduced while still reaching the needed diversion detection probability. This can potentially be achieved by a combination of process monitoring (PM) and non-destructive assay (NDA) in an unattended monitoring approach. The goal of this work is to utilize neural networks—in particular, transformers—and simulated process data to verify the prospect of reducing destructive analysis in favor of unattended monitoring. Our results show that both diversion and off-normal (but non-diversion) scenarios can be detected with up to 99% accuracy.

Introduction

While running a nuclear facility the operator collects large amounts of process monitoring data. This data is already used in a few limited safeguards applications and has been combined with non-destructive assay (NDA) (e.g. gamma measurements on the material) to provide material flow monitoring. However, the current emphasis favors destructive analysis (DA) over process monitoring because the measurement uncertainty is significantly less and therefore it is easier to meet the detection probability metrics required by the specific safeguards approach. This work seeks to improve the power of analysis that can be performed via process monitoring data streams and NDA such that they can be used to supplement or replace destructive assay in some instances.

[*] PNNL-SA-164500

Background

The simulated process data used for this project represents the flow of nuclear materials through a generic aqueous reprocessing plant and was produced by the Separation and Safeguards Performance Model (SSPM) [1]. The model was created with MATLAB Simulink and provides simulated measurements at various control points throughout the facility.

The SSPM generated 900 6480-hour runs of "normal" behavior and 100 runs each of 18 diversion type/rate combinations, resulting in 180 total runs containing data points indicative of diversion. Each of these runs contained 1% random and systematic error and consisted of a measurement of five isotopes (^{134}Cs, ^{137}Cs, ^{154}Eu, ^{241}Am ^{241}Pu) at 20 locations (control points). The diversions were performed at six diversion rates (0.5, 1.0, 1.5, 2.0, 3.0, 3.5) with three types of diversion (mixer, evaporator, and both). For the present work, only Material Balance Area 3 (MBA3) was considered during the collection of the data (Fig. 1). In this MBA only three isotopes, (^{134}Cs, ^{137}Cs, ^{154}Eu) were modeled to produce sufficient signal to make them possible to monitor. As such, analysis consisted of these three isotopes analyzed at the four control points in the MBA.

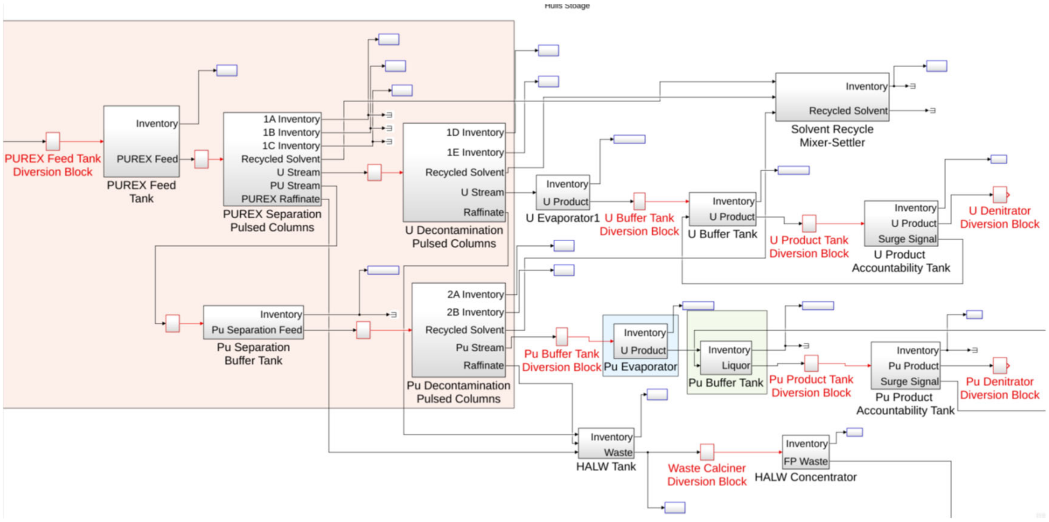


Fig. 1. Material Balance Area 3 (MBA3) of the SSPM [2].

Previous approaches

The previous machine learning approaches relied on a two stage model [2]: First, a long short-term memory (LSTM) [2] model was trained to take a 200-hour sequence of isotope detection history at a single control point and predict the most recent 1-hour isotope detection at a subsequent control point. This prediction would then be compared with the observed output by subtracting the observed output and the predicted output. In the second stage, the history of these differences across the run was fed into an Isolation Forest [4] algorithm to determine if there were indications of diversion [4]. The benefit of this two-stage approach is that both stages are unsupervised; that is, the LSTM model and isolation forest algorithm can be trained on normal behavior of the modeled facility without needing to necessarily model and understand the behavior of the facility during diversion.

Although effective for prediction, there are disadvantages to this approach. Because the LSTM model is not aware of the classifier, the features produced by this model are not optimized for the classification of diversion. The ideal approach would consist of a single model capable of jointly optimizing for discovering discrepancies from normal plant dynamics and classifying them. Second, unsupervised processes have little control over the types of anomalies identified. This is both a strength and weakness compared to supervised methods. If the model is supervised on a manifold of diversive behavior that is a good reflection of the attack surface in the wild, then it may do a superior job of differentiating between off normal behavior that is/isn't in need of additional investigation. However, generating this manifold is of very high difficulty. Thus, strategies employing both supervised and unsupervised methods along with strong explainability tools may form the best alternative. Finally, the LSTM architecture is less capable of extracting features across long input dimensions than more recent architectures, such as Transformers. It would be ideal if the neural structure was maximally sensitive to plant dynamics. Due to these reasons, a single-stage approach utilizing a supervised hierarchical transformer-based architecture was developed.

Methods

Current approach

The approach described in this paper consists of a hierarchical architecture based around the Transformer architecture [5], which applies a mechanism known as self-attention on the input data before feeding the output of this self-attention layer to a conventional feedforward layer. The transformer began as an approach to natural language processing but has recently found enormous success in nearly every major deep learning application, including computer vision [6], which had before been dominated by convolutional neural networks. The attention mechanism compares every point in the input data to every other point and allows the transformer to find longer-range patterns in the data compared to the LSTM model.

Model Architecture

The transformers that compose the model for this approach are arranged in a hierarchical architecture [7]. This architecture consists of a base of three two-layer transformers—one for each isotope (generating features based on data from that isotope at each control point). Prior to feeding the data into these base transformers, each control point vector was prepended with learnable "memory" vectors of length 12, resulting in a total vector length of 512. These memory "prefixes" were inspired by the Memory Transformer [8] and are capable of slightly improving the test set accuracy of the model (Table 1). Following this, each (isotope) group of vectors were concatenated with a learnable latent vector. These latent vectors were inspired by the classification token used in T2T-ViT [9] and BERT [10], which encourages the network to find a more general representation of the input data. These three groups of five 512-length vectors were fed into their respective transformer that extracted features from that isotope alone.

Following the individual isotope transformers is a single "top" transformer. Before reaching this transformer, however, only the feature vectors that had originally corresponded to the latent vectors that were prepended to each isotope group were extracted. These vectors are then concatenated together and topped with a second learnable latent vector and fed into the final "top" transformer. Lastly, the feature vector that had originally corresponded to the second latent vector was once again extracted and provided it to the final linear layer and softmax activation. A simplified diagram of the architecture is provided in Fig. 2.

The key component of the transformer is the attention mechanism. In natural language processing (one of the transformer's first applications), the attention mechanism allows the transformer to learn the dependency between different words in a sentence. In the present case, however, there are two options: (1) Attention can be performed along the length of the sequence in order to calculate the relationship between different data points along the sequence—resulting in an attention matrix (the matrix that represents the relationship between different values along the sequence) of size 512x512; or (2) attention can be performed between each of the individual vectors being fed into each of the transformers, which results in an attention matrix of size 5x5 (in the case of the isotope transformers) or 4x4 (in the case of the top transformer). After testing both methods, the between-vector attention (case 2) resulted in both a significantly faster training time and better performance; therefore, this is the method of attention that was applied throughout the results presented herein.

The internal structure of each transformer (including the isotope transformers and the top transformer) in the architecture is identical, with 512-dimension feedforward layers, 2 heads, 10% dropout layers, and GELU [11] activation. The AdamW optimizer was used with an additional learning rate step decay of 10 epochs at a factor of 0.5 [12]. The loss function used was cross-entropy with label smoothing [13].

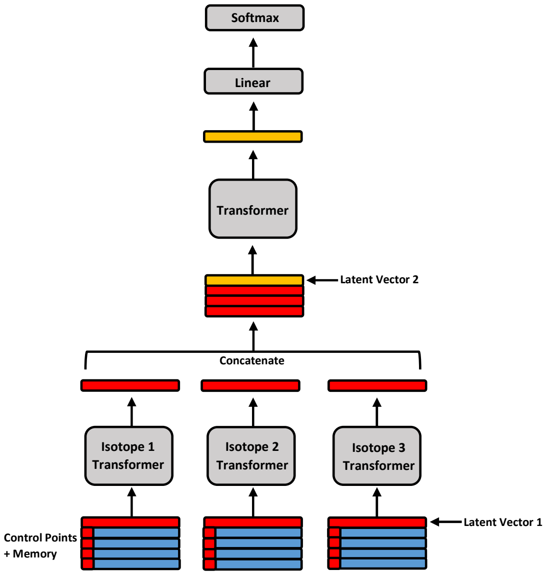


Fig. 2. Model architecture. The red and yellow represent learnable vectors.

Testing on Synthetic Data

Prior to training the model of the data produced by the simulation of the facility, the model was trained and tested using a synthetic data set consisting of small Gaussian peaks embedded in noisy (normal distribution) vectors of the same length as the simulation data at random locations (Fig. 3). The only change made to the model itself was in the final linear layer (where the number of classes were changed to match the classes in the input). The model attempts to locate the position of the hidden Gaussian peak along the length of the vector (i.e. left side or right side) as well as which vector (out of the total 12 vectors) it occurs on, resulting in a total of 25 classes: Class 0 for no Gaussian peak and 24 classes corresponding to one out of 12 vectors and one out of two sides of the vector on which the Gaussian peak is located.

Table 1 provides the test set accuracy produced by experimenting with the removal or addition of the latent vectors and the memory prefixes. “All vectors fed through all layers” represents an architecture in which the original four control points for all three isotopes are fed through their respective isotope transformers. The outputs of these transformers are concatenated to form a group of 12 vectors, which are then fed into the top transformer without the addition of any latent vectors at any point.

Clearly, the addition of the memory prefixes has a significant effect on the ability of the model to generalize. Although the test set accuracy remains low without the memory prefix, the training set accuracy (not shown) continues to rise, indicating overfitting. Without the memory prefixes, the model consistently predicts the correct location of the Gaussian peak along the vector (left/right) but is not able to determine which one of the four vectors within each isotope transformer the peak is on—e.g., if the Gaussian peak was placed on the right-hand side of vector #1 (in the first isotope transformer), the network will be able to predict that it is on the right-hand side of a vector, but it will be unable to determine which vector (within a group of vectors) and will predict that the Gaussian peak is on, say, vector #5. The confusion matrix representing this result is given in Fig. 4(a). Furthermore, whenever only the top latent vector is included in the architecture, the test set accuracy is reduced even further. The cause of this appears to be the inability of the first isotope transformer to correctly propagate the information (Fig. 4(b)). However, if the latent vectors are not used in the model, the predictions become accurate. That is not to say that the latent vectors do not have their use; rather, they encourage the network to produce more general feature representations.

The synthetic data experimentation allowed us to confirm the ability of the model to distinguish between slightly different circumstances—when memory prefixes are included—before training on the facility simulation data. Additionally, the performance of the model was tested with varied memory prefix lengths and have found that only a single scalar is necessary for the network to perform well. Length-12 memory prefixes were chosen specifically due to the divisibility of the total length (512) into different numbers of transformer attention heads. It was also found that the memory prefix does not necessarily need to be a parameter of the network; an initialization of any values (random or otherwise, but not constant) is sufficient for the network to perform well.

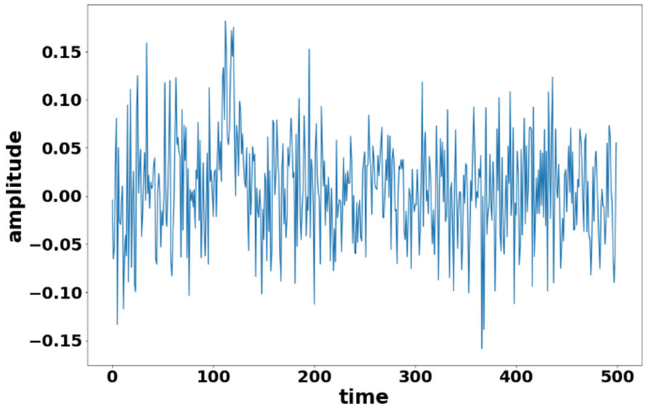
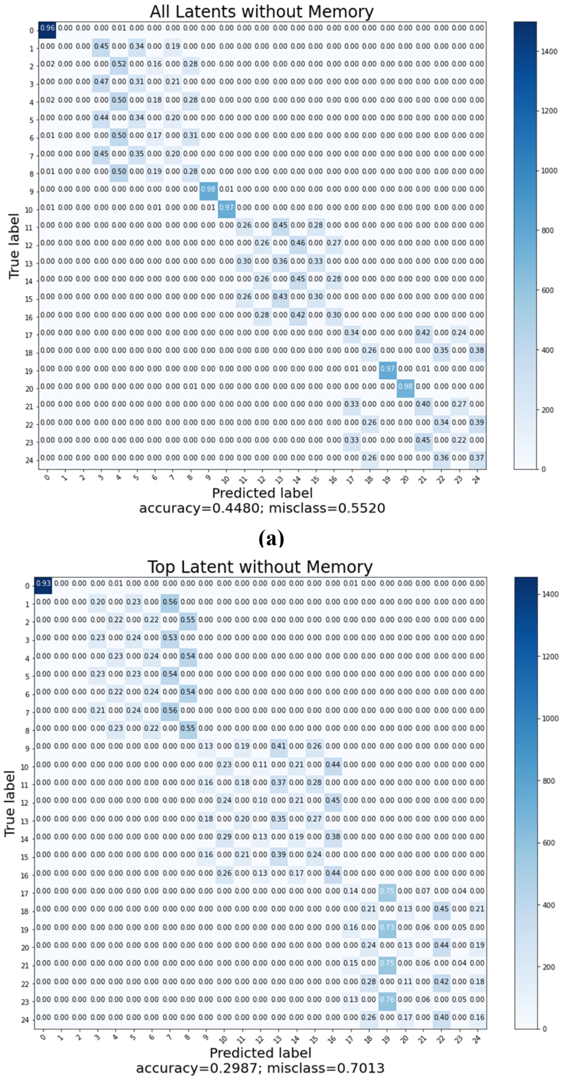


Fig. 3. Example of a single vector of the synthetic data. The embedded Gaussian can be seen on the left side near hour ~110.

Table 1
Results of architecture variations on the synthetic data test accuracy. The accuracy percentage is the best out of 50 total epochs.

Architecture Variation Test Set Accuracy (%) **Synthetic Data**		
Architecture Variation	**With Memory**	**Without Memory**
All vectors fed through all layers	99.0	98.8
Latent vector applied before **isotope** transformers only	98.8	49.6
Latent vector applied before **top** transformer only	99.0	33.0
Latent vectors applied before **all** transformers	98.9	49.7



(a)

(b)

Fig. 4. Per-label prediction results for the Gaussian test data without the memory prefixes. (a) Latent vectors applied before **all** transformers; (b) latent vector applied before **top** transformer only.

Facility Simulation Data Pre-Processing

To make the most efficient use of the data available, the 6,480-hour long sequences were first cut into 500-hour segments (Fig. 5) after the initial sequences were split into 90% training and 10% test sets. The train/test split was done immediately and then segments were generated. For the sequences that contained diversion, the portions indicative of diversion were densely sampled in one-hour increments (the actual sample rate of the data) to maximize the number of training samples with diversion labels. 2000 hours of the sequence immediately following the diversion was skipped in order to allow the sequence to return to normal operation. At the 2000th hour, new sparsely sampled segments were created and labeled as "normal." These normal segments were added to the segments created from the entirely normal sequences to form all of the normal-labeled data. Each segment consisted of data at each hour during the 500 hour duration for each of three isotopes and four control points resulting in the twelve inputs in three groupings depicted in Figure 1.

Table 2 contains the test set accuracies for the architecture variations described in Section 3.2.2. Although the addition of the memory prefixes does not have as dramatic an effect on the network's performance as with the synthetic data, it does appear to help when the latent vectors are used. This is likely due to the network's ability to precisely locate inconsistencies within each isotope group when the memory prefixes are included.

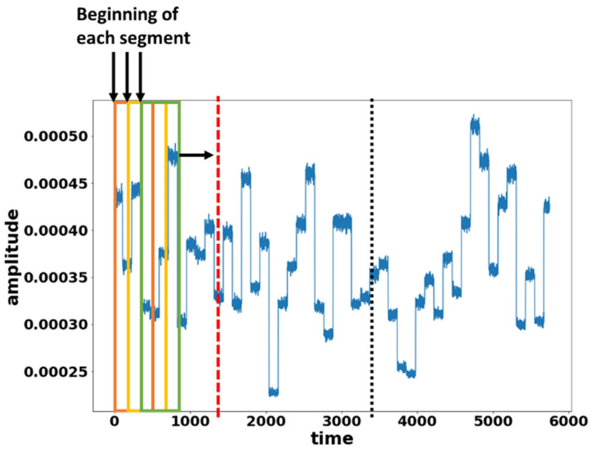


Fig. 5. Initial data processing for the shortest (120 hours) diversion for a sequence taken from one isotope in one control point (beginning at hour 820). The length of time between the beginning of each box (segment) is exaggerated for clarity. They are shifted by only one hour in the dataset.

Table 2
Results of architecture variations on the simulation data test accuracy. The accuracy percentage is the best out of 25 total epochs.

Architecture Variation Test Set Accuracy (%) **Simulation Data**		
Architecture Variation	**With Memory**	**Without Memory**
All vectors fed through all layers	99.1	98.7
Latent vector applied before **isotope** transformers only	99.5	99.0
Latent vector applied before **top** transformer only	99.3	98.1
Latent vectors applied before **all** transformers	99.6	99.0

Results and Discussion

Original Test Set

Although the classifier contains only two possible labels—normal (0) or off-normal (1)—each off-normal (i.e., diversion-positive) 500-hour segment has a secondary label associated with it. These secondary labels represent both the type and the rate of diversion, resulting in 18 different secondary labels (three types of diversion and six different rates). Fig. 6 provides the per-label test set accuracy for these secondary labels. As can be seen, the network is capable of classifying each type of diversion and rate consistently well. The overall test set accuracy of the model was 99.6%.

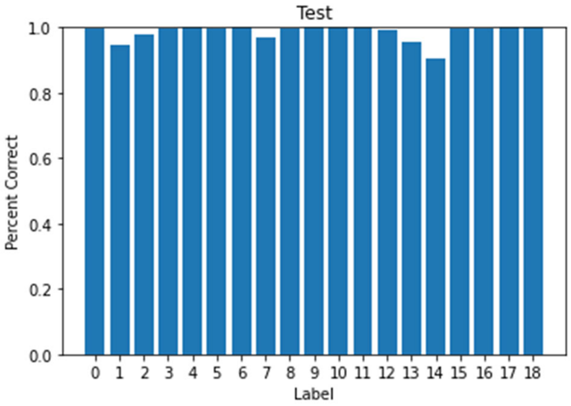


Fig. 6. Per-label test set accuracy.

In order to further assess the network's performance on the test set, the softmax probabilities were plotted as a function of sequence length vs. facility simulation run as an image, as seen in Fig. 7. These sequences were generated by sampling the original runs at a rate of one data point per 25 hours; the vertical axis (Time) represents a segment that was extracted beginning at this time. The normal runs in this test set (the first 1/3 of the image) are perfectly classified as normal for all 90 runs. The 180 runs following the normal runs contain diversion with rates ranging from the first 120 hours to the first 1040 hours. Interestingly, the runs appear to exhibit off-normal behavior for a relatively long period of time after the end of each diversion, and the point at which the model begins to lean towards a normal classification is mostly independent of the length of diversion but rather corresponds to the time at which training inputs were classified as 'normal'. This shows that the plant continues to exhibit off-normal behavior long after the diversion.

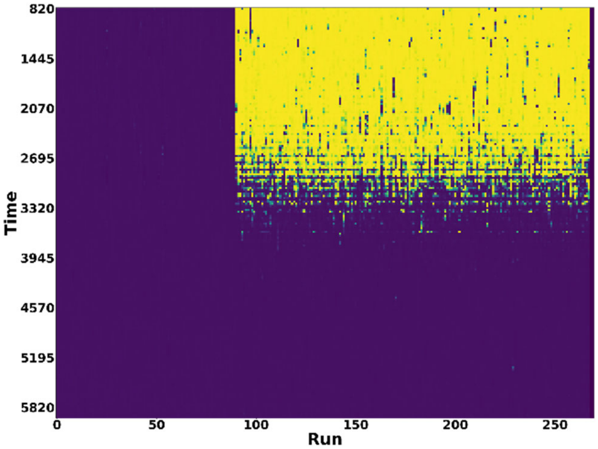


Fig. 7. Sequence length vs. runs for the test data. The left side of the image is comprised of 90 normal runs; the right side is comprised of 180 diversion runs. The color intensity represents the softmax output, with full yellow indicating a prediction of "1" (diversion-positive), purple representing a prediction of "0" (normal), and green representing an uncertain prediction.

Off-Normal Test Set

Although the network performs consistently well on the diversion and normal test data, it does not say anything about the ability of the network to generalize on "upsets" in the facility simulation that it has not trained on. Therefore, a dataset representing off-normal "upsets"—that is, not "normal," but also not necessarily an intentional, known diversion—in the facility simulation was used as a secondary test set. The particular conditions for these off-normal conditions are stated in Table 3. For evaluation on these off-normal conditions, all the runs could be used since there was no training performed. Inputs were generated 25 hours apart (corresponding to evaluation ~1/day) and were 500 hours long. The softmax predictions of the network for each run is shown in Fig. 8.

The figure illustrates the ability of the network to notice off-normal behavior in all 4 diversion cases. Case 1 correctly closely resembles the classification of a diversion as seen in the original test set data. Cases 2-3 show concentrated detection during the time of off-normality, but also express that the system does not fully return to normal during operation. Finally, Case 4 correctly indicates off-normal conditions for the full duration.

Table 2

Conditions induced within the plant simulation to produce the off-normal test cases. See Ref. [1] for more details regarding the simulation.

Conditions for the Off-Normal Test Cases	
	Test Case
Case 1	Short break in PUREX feed output ("PUREX Feed Tank") -- tank continues to accumulate material, but does not output during duration (t=500:550)
Case 2	Leak in decontamination column ("Pu Decontamination Pulsed Columns") -- during the duration 8% of material is lost from the stream and not accounted for (t=600:768)
Case 3	Malfunction in Pu Evaporator ("Pu Evaporator") - increase in liquid solvent fraction in uranium product. Note that as the evaporator is a batch process the change might only be visible for a single batch of material (t=600:768)
Case 4	Incorrect reading on Pu Accountability tank -- results in larger heel left in tank ("Pu Product Accountability Tank") (t=600+)

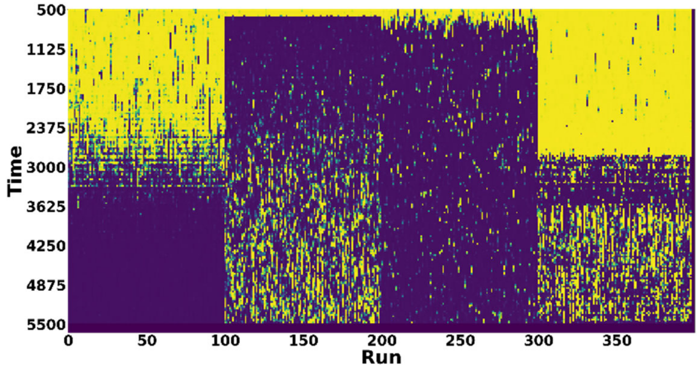


Fig. 8. Sequence length vs. run for the four off-normal cases. There are 100 runs within each case.

Conclusions

The feasibility of using a supervised machine learning model based on the transformer architecture to detect both diversion (on which the network had been trained) and general, off-normal scenarios (which the network had not seen before) in data generated by the Separation and Safeguards Performance Model was shown. The transformer network performed well when testing several variations of the architecture, but the greatest performance is achieved when prepending a "memory" vector to the beginning of each segment before they are fed into the model. It was found that the effect of the memory prefix is even more significant when testing the architecture on a synthetic test data set containing 25 position-specific classes.

While using the transformer network to further analyze the original test set consisting of both normal and diversion scenarios, the network predicted that off-normal behavior continued within the simulated facility long after the initial diversion. Similarly, on the off-normal (non-diversion) data, the network not only predicts the upset behavior, but also finds that off-normal behavior may continue throughout the entire simulated sequence—or become even more significant as time goes on.

Acknowledgments

This paper is an account of work funded by the U.S. DOE/NNSA Office of Nonproliferation and Arms Control, whose support is gratefully acknowledged.

References

[1] Cipiti, Benjamin B., and Nathan Shoman. "Bulk Handling Facility Modeling and Simulation for Safeguards Analysis." Science and Technology of Nuclear Installations 2018 (2018).

[2] Nathan Shoman, and Ben Cipiti"Advances in Machine Learning for Safeguarding a PUREX Reprocessing Facility." INMM ESARDA Joint Annual Meeting (2020)

[3] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9, no. 8 (1997): 1735-1780.

[4] Liu, Fei Tony, Kai Ming Ting, and Zhi-Hua Zhou. "Isolation forest." In 2008 eighth ieee international conference on data mining, pp. 413-422. IEEE, 2008.

[5] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. "Attention is all you need." arXiv preprint arXiv:1706.03762 (2017).

[6] Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani et al. "An image is worth 16x16 words: Transformers for image recognition at scale." arXiv preprint arXiv:2010.11929 (2020).

[7]Zichao Yang, et al., *Hierarchical Attention Networks for Document Classification,* http://www.cs.cmu.edu/~./hovy/papers/16HLT-hierarchical-attention-networks.pdf

[8] Burtsev, Mikhail S., Yuri Kuratov, Anton Peganov, and Grigory V. Sapunov. "Memory transformer." arXiv preprint arXiv:2006.11527 (2020).

[9] Yuan, Li, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. "Tokens-to-token vit: Training vision transformers from scratch on imagenet." arXiv preprint arXiv:2101.11986 (2021).

[10] Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

[11] Hendrycks, Dan, and Kevin Gimpel. "Gaussian error linear units (gelus)." arXiv preprint arXiv:1606.08415 (2016).

[12] Loshchilov, Ilya, and Frank Hutter. "Decoupled weight decay regularization." arXiv preprint arXiv:1711.05101 (2017).

[13] Zhang, Chang-Bin, Peng-Tao Jiang, Qibin Hou, Yunchao Wei, Qi Han, Zhen Li, and Ming-Ming Cheng. "Delving Deep into Label Smoothing." arXiv preprint arXiv:2011.12562 (2020).