Expert Systems With Applications 261 (2025) 125521

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

APT: Alarm Prediction Transformer

Nika Strem [a,*], Devendra Singh Dhami [a,b], Benedikt Schmidt [c], Benjamin Klöpper [c], Kristian Kersting [a,b]

[a] TU Darmstadt, Hochschulstrasse 1, Darmstadt, 64289, Germany
[b] Hessian Center for Artificial Intelligence (hessian.ai), Darmstadt, Germany
[c] ABB AG, Kallstadter Straße 1, Mannheim, 68309, Germany

ARTICLE INFO

Keywords:
Machine learning
Deep learning
Industrial processes
Alarm management
Multimodal transformer
Multimodal fusion

ABSTRACT

Distributed control systems (DCS) are essential to operate complex industrial processes. A major part of a DCS is the alarm system, which helps plant operators to keep the processes stable and safe. Alarms are defined as threshold values on individual signals taking into account minimum reaction time of the human operator. In reality, however, alarms are often noisy and overwhelming, and thus can be easily overlooked by the operators. Early alarm prediction can give the operator more time to react and introduce corrective actions to avoid downtime and negative impact on human safety and the environment. In this context, we introduce Alarm Prediction Transformer (APT), a multimodal Transformer-based machine learning model for early alarm prediction based on the combination of recent events and signal data. Specifically, we propose two novel fusion strategies and three methods of label encoding with various levels of granularity. Given a window of several minutes of event logs and signal data, our model predicts whether an alarm is going to be triggered after a few minutes and, if yes, it also predicts its location. Our experiments on two novel real industrial plant data sets and a simulated data set show that the model is capable of predicting alarms with the given horizon and that our proposed fusion technique combining inputs from different modalities, i. e. events and signals, yields more accurate results than any of the modalities alone or conventional fusion techniques.

1. Introduction

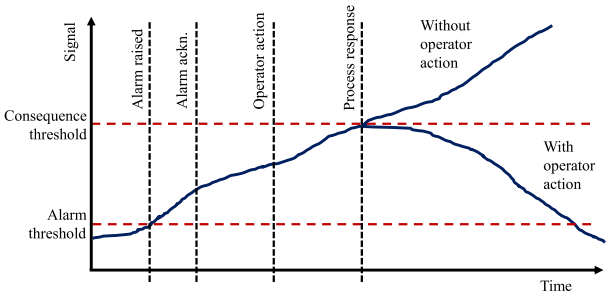
In modern industries, with increasingly comprehensive automation, plant operators are facing an increasingly complex scope of tasks, including monitoring numerous process indicators to ensure normal operation and process safety. Yet the number of measurements to track and assess is overwhelming for a human, therefore, in case of deviations, the process monitoring system triggers alarms to indicate an impending critical situation, equipment malfunction, process deviation, or abnormal condition requiring a timely response from the operator (IEC 62682 Management of Alarm Systems for the Process Industries, 2014). Alarms are defined based on thresholds that may not be exceeded by sensor measurements, such as temperature, flow, level or pressure. Alarm thresholds are configured in such a way as to notify the operator early enough to allow for corrective actions, yet not too early to avoid noisy alarms caused by temporary fluctuations (Fig. 1(a)). When an alarm is raised, the operator must take corrective action, such as opening or closing a valve, or changing a setpoint value to bring the process back to normal conditions. If the operator fails to respond timely to prevent further deterioration of the situation, this can lead to failures, damage of equipment, downtime and hazards to the employees (Fig. 1(b)).

Whereas incipient changes in the system leading to a state where an alarm is triggered cannot be noticed by a human, a machine learning model trained on historic data can potentially capture characteristic patterns, interactions and dependencies among various signals and events. Such a model can be used to predict alarms before they occur to give an operator more time to resolve the issue and additionally help identify the cause of the problem. Real-world industrial data is often multivariate and complex. To provide the operator with early warning, unlike studies that predict alarms based on past alarms or concentrate on a particular alarm type, we attempt to predict an alarm when there have not been any recent alarms and the situation seems normal.

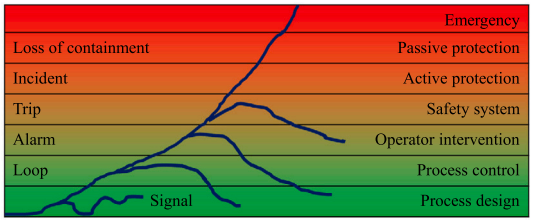
Within the context of distributed control systems, there are two major data types that could be used for data-driven early alarm prediction: *signals* and *events*. Signals include measurements of physical quantities like temperatures, flows, pressures; setpoint and output values of control loops; and information about actuators like rotations per minute or

* Corresponding author.
E-mail addresses: nika.strem@cs.tu-darmstadt.de (N. Strem), devendra.dhami@cs.tu-darmstadt.de (D.S. Dhami), benedikt.schmidt@de.abb.com (B. Schmidt), benjamin.kloepper@de.abb.com (B. Klöpper), kersting@cs.tu-darmstadt.de (K. Kersting).

https://doi.org/10.1016/j.eswa.2024.125521
Received 17 January 2023; Received in revised form 4 October 2024; Accepted 5 October 2024
Available online 15 October 2024



(a) Alarm response timeline (adapted from (IEC 62682 Management of Alarm Systems for the Process Industries, 2014))



(b) Layers of protection (adapted from (Stauffer & Clarke, 2016))

Fig. 1. Alarms notify the operator of an equipment failure or process deviation requiring an intervention to restore normal operation. Alarm thresholds are defined taking into account such factors as operator response time, process schedule and response time, as well as severity of potential consequences. Without operator's response, an anomaly can result in equipment damage and downtime, or even major hazards or a plant shutdown.

valve positions. Events include changes automatically registered in the system or introduced by an operator, binary changes, setpoint changes, alarms with different severity levels, etc. It is important to notice that only limited information is displayed to operators to avoid information overload.

Existing machine learning based approaches to early alarm prediction analyze signals and events in an isolated fashion (see Section 2.1). At the same time, considering that most real world data comprises more than a single modality, there has recently been a surge in methods that belong to multimodal machine learning, especially for high impact real-world problems such as hurricane forecasting (Boussioux, Zeng, Guénais, & Bertsimas, 2022), drug-drug interaction prediction (Dhami, Yan, Kunapuli, Page, & Natarajan, 2021), healthcare (Tiulpin et al., 2019) and human–robot collaboration (Liu, Fang, Zhou, Wang, & Wang, 2018), to name a few. In particular, Transformers (Vaswani et al., 2017) have revolutionized the performance of deep neural networks and are used in a wide variety of real-world applications: from machine translation (Devlin, Chang, Lee, & Toutanova, 2018; Radford et al., 2019) to protein prediction (Nambiar et al., 2020) and even playing chess (Noever, Ciolino, & Kalin, 2020). Transformers have also been extended to multimodal setting (Rahman et al., 2020) but most implementations are focused on combining the standard modalities such as image, text and audio (Akbari et al., 2021; Chen, Guhur, Schmid, & Laptev, 2021). Despite the success of multimodal machine learning methods and Transformers in particular in other domains, to the best of our knowledge, there have been no attempts yet to model multimodal industrial data using the Transformer architecture.

To this end, we introduce **A**larm **P**rediction **T**ransformer (APT), a multimodal Transformer-based model for early alarm prediction based on the combination of recent events and signal data. Given a window of several minutes of event logs and signal data, the model predicts whether an alarm is going to be triggered after the next few minutes. In addition, while analyzing data coming in from an entire plant, the model learns to identify the problematic area within the plant where the alarm is predicted to happen and also predicts the alarm location. In APT, we compare three alternative strategies of combining inputs from different modalities: 'skip' fusion, whereby input modalities are transformed to a uniform hidden dimensionality separately and passed to the decoder directly, without applying self-attention; 'early' fusion, whereby the initially transformed inputs are combined and passed through a single Transformer block; and 'late' fusion, whereby separate self-attention Transformer blocks are used for each modality and their outputs are combined. Going beyond these classical fusion types, we propose a novel **'hybrid'** fusion, whereby early and late fusion encoders are combined in a single block. Whereas either early or late fusion may prove more efficient for different datasets and applications, hybrid fusion could achieve on par or better performance without preliminary fine-tuning. Finally, a decoder is applied to predict the alarms sequentially at various levels of granularity. APT is validated on a real-world customer use case.

Overall, we make the following important contributions:

C1. *We present APT, the first work on using Transformers in a multimodal setting for real industrial data thereby paving the way for industrial transformation.*
C2. *We propose to predict alarm labels sequentially, by encoding them in three novel ways, namely (a) as characters, (b) as 'morphemes' and (c) as entire tags.*
C3. *We propose a novel fusion strategy and show that it achieves on par or higher performance than the commonly used early and late fusion methods.*
C4. *We show that APT is effective in dealing with heavily unbalanced datasets with 100 s of classes.*

2. Multimodal learning in industrial settings and academia

Machine learning techniques are widely studied in application to industrial tasks, such as alarm prediction, however, they mostly rely on unimodal data.

2.1. Alarm prediction techniques

Data-driven approaches to alarm prediction commonly use signal data. For instance, Langone, Alzate, Bey-Temsamani, and Suykens (2014) train a nonlinear autoregressive model for temperature prediction. Based on its forecast, a binary classifier predicts future alarms. Similarly, Villalobos, Suykens, and Illarramendi (2021) forecast sensor measurements with an LSTM model and apply a ResNet classifier to predict alarms based on the forecast. Proto et al. (2019-07) predict alarms using a Random Forest or Gradient Boosted Tree classifiers using summary statistics over process variables as tabular inputs. In a similar vein, Li, Qian, Parikh, and Hampapur (2013-10) train a customized Support Vector Machine to predict bearing related alarms from statistics over sensor measurements. Koltsidopoulos Papatzimos, Thies, and Dawood (2019) predict wind turbine alarms based on wind speed distribution analysis. Chatterjee and Dethlefs (2020) use a transformer model to predict an alarm class from sensor measurements of wind turbines. In other studies, next alarm is predicted based on previous alarm records, for example, Zhu, Wang, Li, Gao, and Zhao (2016) convert alarm sequences into n-grams and apply maximum likelihood estimation to predict the next alarm. Cai, Palazoglu, Zhang, and Hu (2019) convert alarm sequences into word embeddings and predict the next alarm using an LSTM network. Wang and Liang (2020) train binary LSTM-based classifiers per each alarm type and predict the next alarm using alarm clustering and model voting. In relying on a sequence of already triggered alarms however, such studies lie out of the scope of early alarm prediction.

To the best of our knowledge, multimodal learning has not yet been used to solve the task of early alarm prediction.

2.2. Fault detection techniques

In a related task of fault detection, several studies can be found where modalities are combined. Inceoglu, Aksoy, Cihan Ak, and Sariel (2021) use a multimodal classifier for failure detection. The model uses early fusion to combine RGB and depth frames in convolutional and convLSTM layers, then late-fuses the output via concatenation with that of convolutional layers for audio data, then applies fully connected layers and produces a binary output. In a similar fashion, Li et al. (2021) use a stacked denoising autoencoder to extract audio features and a CNN to process images. The concatenated outputs are passed through linear and softmax layers to predict a fault. Yang, Baraldi, and Zio (2021) use separate CNNs for images and text records and a fully connected network for structured maintenance data. The concatenated outputs are passed through a regression layer to output degradation level. Limoyo, Ablett, and Kelly (2022) late-fuse outputs from image and signal CNNs in a GRU network to predict real-valued 2D position commands. The majority of approaches however also use unimodal data, namely, process variables: (Datong, Yu, & Xiyuan, 2009-05; Di Lello, Klotzbucher, De Laet, & Bruyninckx, 2013; Fadzail, Mat Zali, Mid, & Jailani, 2022; Giurgiu & Schumann, 2019; Helbing & Ritter, 2018-12-01; Langone, Cuzzocrea, & Skantzos, 2020-11-01; Li, Wang, & Wang, 2017; Lomov, Lyubimov, Makarov, & Zhukov, 2021; Lucke, Stief, Chioua, Ottewill, & Thornhill, 2020-01-01; Reinartz, Kulahci, & Ravn, 2021; Zhao, Hu, Ai, Hu, & Meng, 2018-03).

2.3. Multimodal learning outside of industrial context

Outside of the industrial context, multimodal learning is studied much more extensively. Most studies rely on the state-of-the-art Transformer models (Vaswani et al., 2017), which successfully capture dependencies between data points regardless of their distance in the input or output sequences. In particular, multimodal Transformers pretrained on large datasets (with millions of samples) have become popular, especially in the audiovisual domain. Various models rely on different types of fusion and pretraining strategies.

ViLBERT (Lu, Batra, Parikh, & Lee, 2019) is a joint model for learning task-agnostic visual grounding from paired visiolinguistic data with co-attention pretrained on multimodal tasks. LXMERT (Tan & Bansal, 2019) consists of two unimodal and one cross-modal encoders, pretrained on multimodal tasks: masked language modeling, masked object prediction (feature regression and label classification), cross-modality matching, and image question answering. Sun, Baradel, Murphy, and Schmid (2019) extend an existing language Transformer with a video encoder and pretrain simultaneously on losses from individual modalities and a cross-modal noise contrastive estimation loss. MuLT (Tsai et al., 2019) is built from multiple stacks of pairwise and bidirectional crossmodal attention blocks that directly attend to low-level features (without self-attention). They use six pairs of crossmodal Transformers, plus one Transformer per modality to merge two corresponding crossmodal blocks. The model has no decoder and is not pretrained. OmniNet (Pramanik, Agrawal, & Hussain, 2019) combines modalities using gated multi-head attention and is trained simultaneously on several unimodal and multimodal tasks. CLIP (Radford et al., 2021) jointly trains image and text encoders to align samples using a contrastive loss. At test time the learned text encoder synthesizes a zero-shot linear classifier. In UFO (Wang et al., 2021), a single Transformer network is used as the image encoder, the text encoder, or the fusion network in different pretraining tasks with the image-text contrastive loss, image-text matching loss, and masked language modeling loss (with one pretraining loss randomly sampled per epoch). UniT model (Hu & Singh, 2021), consisting of modality-specific encoders and a shared decoder with task-specific output heads, is jointly trained end-to-end on all tasks. Cho, Lei, Tan, and Bansal (2021) extend text encoders of existing language Transformers with image region embeddings and pretrain on multimodal VL tasks. OFA (Wang et al., 2022) is an encoder–decoder framework pretrained on uni- and multimodal tasks where output modalities are represented in a unified space using a unified output vocabulary.

3. Alarm Prediction Transformer (APT)

In modern industries, thousands of signals are continuously monitored to ensure stable operation. When a signal crosses a pre-defined threshold, an alarm is triggered to indicate a potential problem to the operator, yet the existing alarm configurations tend to generate too many false positive alarms, thus confusing the operator. We propose a data-driven method for early alarm prediction which, based on the combination of recent sensor readings and events, detects anomalous patterns in the data and gives the operator an early warning, thus reducing the amount of noise and allowing for more time to fix the issue. The method leverages state-of-the-art deep learning techniques, extended through a novel multimodal fusion technique combining two complementary modalities, and a sequential prediction approach.

3.1. Early alarm prediction methodology

In industrial processes, events and sensor measurements are continuously logged by the control system. Anomalous conditions are indicated to the operator by triggering alarms when predefined thresholds are crossed. Assuming that deviations in the process can start manifesting themselves in events and in the interaction of various signals before a threshold of a specific process variable is reached, the approach is to detect such deviations in a data-driven fashion and give the operator an early warning. To achieve this, at every point in time t:

1. A window of N minutes of most recent events and signal data from $t - N$ to t is fed into the model's encoder — Fig. 2 (1).
2. The model, trained on historic data, makes a prediction M minutes into the future including a binary output (whether and alarm is going to fire at time $t + M$) and a multiclass output (at which tag an alarm is going to be triggered) — Fig. 2 (2).
3. If an alarm is predicted, an early warning is shown to the operator so that he can take corrective actions in time — Fig. 2 (3).

Based on preliminary experiments as well as discussions with the customer, N was chosen to be 15 min as a time window sufficiently long to contain useful information, whereas M was fixed at 5 min as a prediction horizon which both enables high prediction accuracy and allows the operator to take corrective actions to prevent an actual alarm from being triggered. Both the input window N and the prediction horizon M can be chosen depending on the dataset and the properties of a given process.

Problem Statement. Predict an alarm in conditions which appear normal to the operator, that is, in the absence of other alarms.

Without an early warning, an upcoming alarm would fire unexpectedly for the operator. Therefore, to train the model, samples are selected in such a way that none of the data windows contain previous alarms in the past 20 min. The model is trained on a dataset consisting of alarm-free time windows followed by alarms at time $t + M$, as well as alarm-free time windows that are not followed by alarms. Each time window contains two input modalities: events and sensor measurements, and is followed by a label sequence indicating the presence or absence of alarms at time $t + M$ as well as the alarm tag.

3.2. Multimodal learning

We introduce **A**larm **P**rediction **T**ransformer (APT), a multimodal attention-based model for early alarm prediction based on the combination of recent events and signal data. Existing approaches to alarm prediction rely either on signals or events separately (see Section 2.1). Encouraged by the recent advances in multimodal learning (see Section 2.3), we propose to combine two complementary modalities to

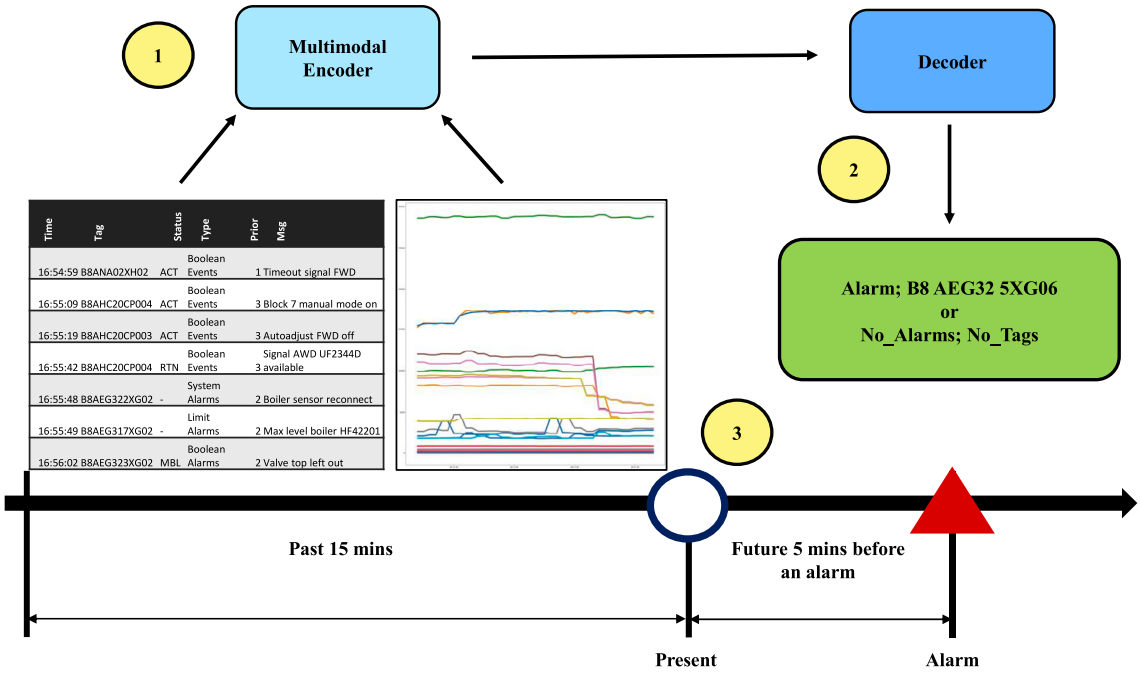
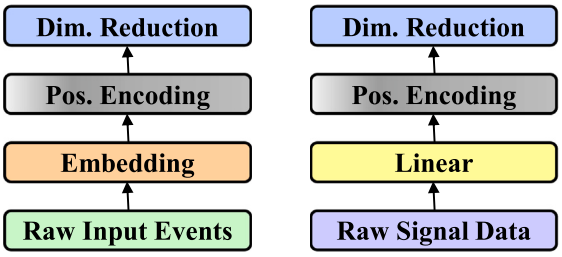


Fig. 2. Input and output modalities. The encoder creates a representation of the past 15 min of events and signal data, which is used by the decoder to forecast an alarm 5 min into the future.



(a) Input layers for events (b) Input layers for signals

Fig. 3. Input layers of the model for the initial transformation of the two modalities into the hidden space.

leverage the model's representation learning capacity and maximize prediction accuracy. To predict alarms across the entire plant and also identify the specific tag where an alarm would fire, a model must learn a rich representation of the underlying industrial process, capturing not only patterns in the individual process variables but also their interactions with other signals and events.

The two modalities include signal data and events. Signal data is represented as continuous multivariate time series, whereas events are logged as structured entries indicating state changes (each containing a tag, status, event type, priority, operator message, etc.), which are concatenated and treated as text. Both signal and event data have timestamps, however, the two modalities are not aligned: unlike process variables, which can be resampled at a desired rate, events are unevenly distributed over time, therefore, samples are created using sliding windows. For each alarm, time windows are selected starting 20 min before an alarm and ending 5 min before it, and balanced with randomly sampled time windows of the same duration not followed by alarms. Since the goal is to learn process representation in a data-driven way and develop a scalable solution which can be directly applied to a different dataset or a different process (see Section 4.1.2), all events and process variables are treated equally and sliced into windows on the same timestamps.

Both modalities undergo necessary preprocessing. Signal data is resampled, imputed and scaled, then sliced based on timestamps. Events are selected based on timestamps and filtered, then useful attributes, e.g., tag, status, type, priority, and operator message, are kept, while timestamps and uninformative columns are removed, and string representations of event entries are tokenized (unlike natural language, event attributes contain abbreviations, special terms and symbols, therefore no lemmatization or similar procedures are necessary and entries are simply split into tokens on whitespace). Sequences of tokens representing events within the same time windows are concatenated, then either padded or left-trimmed to the length of 300 (keeping the most recent ones).

To enable multimodal processing, time windows of recent events and sensor measurements first undergo a series of modality-specific transformations in the input layers of APT (Fig. 3). In the first layer, the window of scaled process variables undergoes a linear transformation, while the sequence of event tokens is passed through an embedding layer such that both modalities are projected into the same hidden dimension D. The input features of each sample can be denoted as $X_{\{S,E\}} \in \mathbb{R}^{T_{\{S,E\}} \times D_{\{S,E\}}}$, where T denotes the number of timestamps within a window for signal data S and the number of tokens within the concatenated sequence of events E (sequences of event tokens are trimmed to the length of 300). Both the embedding layer for events and the linear layer for signal data have a hidden dimension of 512. A learnable positional encoding is added to the transformed inputs of both modalities separately. Next, a linear transformation is applied to both modalities to reduce dimensionality to 128. These initial layers preparing events and process variables for multimodal processing can be seen in Figs. 3(a) and 3(b), respectively.

3.3. Overall transformer architecture

We treat the alarm forecasting task as a sequence-to-sequence learning problem and tackle it using an encoder–decoder Transformer model shown in Fig. 4. The encoder projects the multimodal input sequence of events and continuous multivariate signal data from the past several minutes to a joint hidden representation, which is then fed into the decoder. The decoder autoregressively generates a sequence of

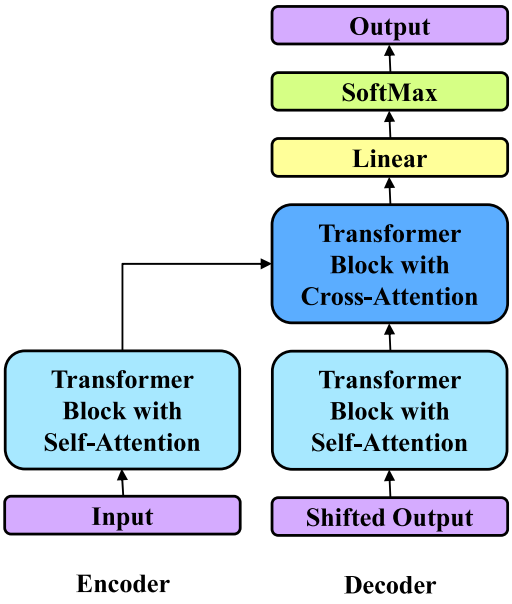


Fig. 4. Model architecture: encoder–decoder.

output tokens representing the presence or absence of an upcoming alarm and a specific alarm tag indicating where in the system the alarm is expected to be triggered. At each step, decoder relies both on the representation from the encoder and on the previously generated symbols.

APT uses Transformer architecture (Vaswani et al., 2017), the state-of-the-art machine learning model relying entirely on self-attention to compute representations of its input and output sequences, without using RNNs or convolutions (which makes Transformers faster). By attending over each position in a sequence, Transformers can learn long-range dependencies, thus overcoming the known limitation of recurrent networks (Hochreiter, Bengio, Frasconi, Schmidhuber, et al., 2001).

The initial layers of APT are followed by a Transformer block, which in its turn consists of two Transformer layers. Each Transformer layer, as shown in Fig. 5 includes a four-headed attention sublayer, and a fully connected feed-forward sublayer, each followed by a residual connection. Following previous works (Klein, Kim, Deng, Senellart, & Rush, 2017; Vaswani et al., 2018), APT uses pre-normalization where layer normalization precedes a Transformer layer.

The model includes three Transformer blocks: a self-attention block in the encoder, a self-attention block in the decoder and a cross-attention block in the decoder (Fig. 4). Correspondingly, attention layers of these blocks take different inputs:

1. In a self-attention Transformer block in the encoder, encoder inputs (i. e. windows of events and process variables) serve as inputs to the self-attention layer — Fig. 6(a).
2. In a self-attention Transformer block in the decoder, decoder inputs (i. e. the auxiliary beginning-of-sequence token (<bos>) followed by the previously generated output sequence designating the alarm) serve as inputs to the self-attention layer.
3. In a cross-attention Transformer block in the decoder, decoder inputs along with the encoder outputs serve as inputs to the cross-attention layer — Fig. 6(b).

A single attention layer applies a scaled dot product operation to the inputs as shown in Figs. 6(a) and 6(b), followed by a linear projection. In the final decoder layer, the outputs of the cross-attention Transformer block are passed through a linear layer, followed by a softmax,

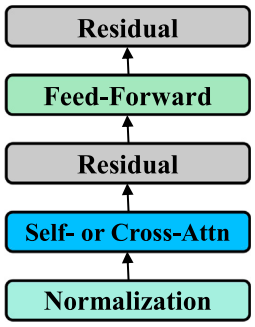


Fig. 5. Transformer block.

and are finally mapped to the output symbols. At inference time, for each sample, the prediction is generated sequentially, predicting one output token at a time: while the encoder output representing events and process variables remains fixed, the decoder output sequence representing the alarm and its tag is extended through a newly generated token at each step, and this extended sequence is fed into the decoder self-attention Transformer block to generate the next token until the predefined sequence length is reached. A full pass through the model is described in detail in Algs. 1 and 2.

3.4. Multimodal encoder

To combine the input modalities (events and process variables), the encoder can incorporate one of four various modality fusion types, namely, the state-of-the-art early and late fusion, as well as the novel 'skip' and hybrid fusion. The traditional early and late fusion as well as the novel 'skip' fusion are implemented as baselines whereas the proposed APT model incorporates the novel hybrid fusion. The different fusion types are implemented as follows:

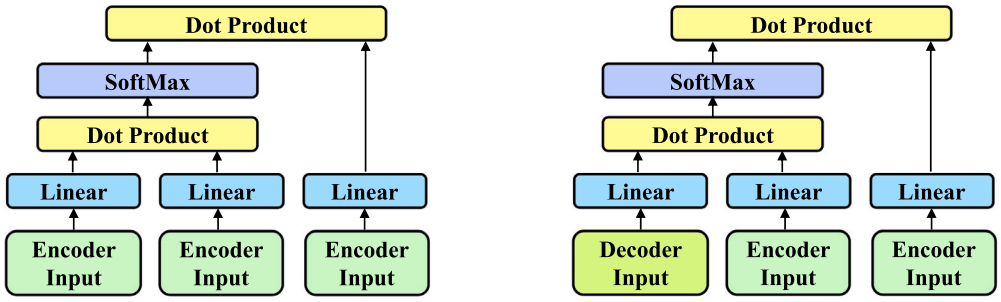
1. In ***early fusion***, the transformed inputs from both modalities are concatenated and passed through a single Transformer block (Fig. 7(a)).
2. In ***late fusion***, signal data and events are passed through separate Transformer blocks and concatenated after (Fig. 7(b)).
3. ***'Skip' fusion*** implies a simple concatenation of the transformed inputs from both modalities, the result of which is passed to the decoder directly, without self-attention, like a residual connection (Fig. 7(c)).
4. ***Hybrid fusion*** is a combination of the early and late fusion: on the one hand, signal data and events are concatenated and passed through a single Transformer block, on the other, the same inputs are passed through separate Transformer blocks and concatenated after. Then, the outputs of both the early-fused and the late-fused blocks are concatenated and passed to the decoder (Fig. 7(d)).

Whereas either early or late fusion may achieve higher accuracy depending on the dataset, the hybrid fusion has the potential to combine the advantages of both early and late fusion.

A forward pass through the encoder with hybrid fusion is captured in Alg. 1.

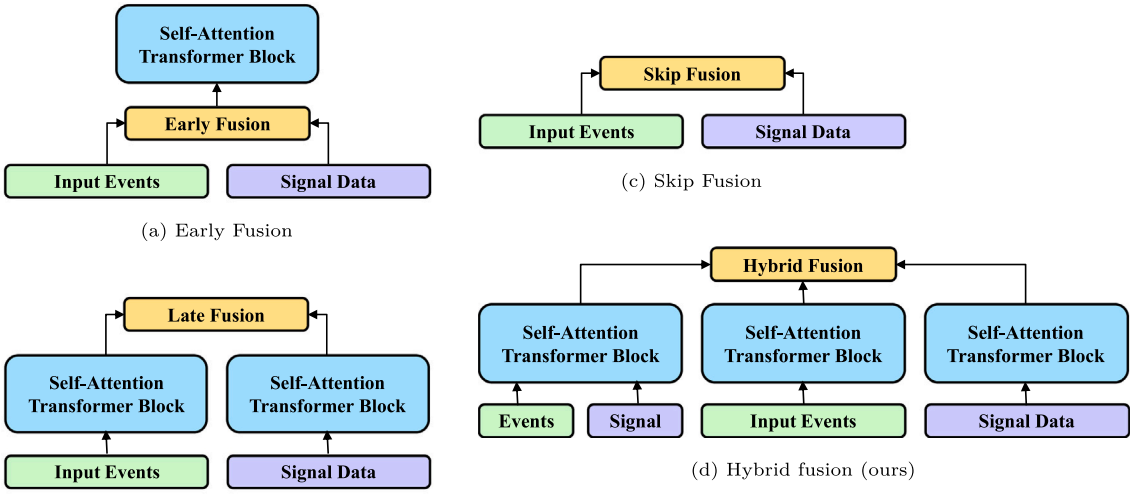
3.5. Alarm decoder

In addition to the binary prediction of whether an alarm would occur within the given horizon, the model also predicts its tag, i. e. the specific component where the alarm would fire. Tags are encoded according to KKS, the identification standard for power stations (Essen, 2021), as an alphanumeric string designating the hierarchy of a component's location in the plant: the plant, function, equipment



(a) Encoder self-attention (b) Decoder cross-attention

Fig. 6. Attention layers of the encoder and decoder. Encoder self-attention computes a representation of the input sequence by relating different positions within the sequence. Decoder attention layer is similar to that of the encoder, but with decoder inputs 'attending' over the encoder outputs.



(a) Early Fusion (c) Skip Fusion

(b) Late Fusion (d) Hybrid fusion (ours)

Fig. 7. Fusion types: the state-of-the-art early and late fusion and the novel skip and hybrid fusion (used in APT).

and component. To enable the model to implicitly learn this structure, the prediction is made sequentially. The target sequence begins with a binary identifier `Alarm`/`No_alarms`, followed by the token `No_tag` in case of `No_alarms`, else by the alphanumeric tag representation (e.g., `XOHNA7OFQO13XH52`). Based on the domain knowledge about the structure of tag labels, we propose to encode them at one of three levels:

1. ***Entire tags:*** As a baseline option, each tag is encoded as a single token (e.g., `Alarm XOHNA7OFQO13XH52`). In this case, the sequence length is minimal (2 without <bos> tokens), but the number of classes is maximal (108 or 200 in our datasets). On the one hand, small sequence length is beneficial in that the decoder would be less prone to errors at test time that could be caused by potentially erroneous outputs in the already predicted part of the sequence. On the other, it is hard to capture the patterns for a high number of classes given their highly imbalanced distribution.
2. ***Characters:*** As an opposite approach, the tags are split into characters, each of them being treated as a separate token (e.g., `Alarm X O H N A 7 O F Q O 1 3 X H 5 2`). This permits to reduce the number of classes to be predicted at each time step to a minimum (43), however, the sequence length is maximal with this option (tag length for alarms equals 16 characters). This can help the model learn the character-level n-gram structure of tags correlated with input windows. At the same time, this can result in prediction of strings of characters which are likely to co-occur but do not constitute a valid tag label.
3. ***Morphemes:*** Finally, the tags are split into four parts corresponding to the plant, function, equipment and component (e.g., `Alarm XO HNA7O FQO13 XH52`). The motivation for this is that it could be easier for the model to capture correlations at different hierarchical levels rather than tags as a whole, given the limited amount of data and its unbalanced distribution. At the same time, this allows to reduce the probability of generating invalid tags. The number of classes in this case is slightly lower than that of the entire tags (103 or 166 for the given datasets), and the sequences remain reasonably short (5).

Overall, these three options represent a trade-off between the number of classes and the length of the sequence to be predicted. The decoder generates its prediction for each subsequent token based on the hidden representation produced by the multimodal encoder and the previously generated tokens, beginning with the auxiliary beginning-of-sequence token (<bos>). Given that, unlike with natural language generation, where the outputs need to be not only semantically plausible, but also stylistically varied, in case of alarm prediction the highest priority is fidelity to ground truth, tokens are generated greedily, at each step selecting the one with highest probability.

A forward pass through the decoder is described in Alg. 2.

3.6. Training

During training, for each window of events and signals, the model iteratively outputs a sequence of tokens representing an alarm prediction. Given the ground truth labels, cross-entropy loss is calculated for

Algorithm 1 Encoder Pass

Input: $X_e, \hat{X}_a \in V^*$, sequence of token IDs of input events; $X_s \in R^*$, input sequence of signals
Output: enc, encoding of the input sequences of events and alarms
Hyperparameters: $l_e, l_s, d_e, d_f, n_e, n_s \in N$, lengths of input sequences of event tokens and signals; embedding depth and depth of the feed-forward layers; number of encoder layers and signals
Parameters:
$W_e^e \in R^{d_e \times |V|}$, $W_{pe}^e \in R^{d_e \times l_e}$; $W_e^s \in R^{d_e \times n_s}$, $W_{pe}^s \in R^{d_e \times l_e}$, embedding and positional encoding matrices for events and signals
For $l \in [n_e]$: $W_l^{sAtE}, W_l^{sAtS}, W_l^{sAtES} \in R^{d_e \times d_e}$, $W_l^{ffIE}, W_l^{ffIS}, W_l^{ffIES} \in R^{d_f \times d_e}$, $W_l^{ffOE}, W_l^{ffOS}, W_l^{ffOES} \in R^{d_e \times d_f}$, parameters of encoder self-attention *(sAt)* and inner *(ffI)* and outer *(ffO)* feed-forward layers for individual *(E, S)* and fused *(ES)* events and signals

function ATTN(Q, K, V, W)
  $[Q, K, V] \leftarrow [Q, K, V] \times W$
  return softmax$(\frac{QK^T}{\sqrt{d_e}})V$
end function

Encoding:
$enc_e \leftarrow W_e^e X_e + W_{pe}^e X_e$
$enc_s \leftarrow W_e^s X_s + W_{pe}^s X_e$
$enc_{es} \leftarrow [W_e^e X_e + W_{pe}^e X_e, W_e^s X_s + W_{pe}^s X_e]$
for $l \in [n_e]$ **do**
  $enc_e \leftarrow$ ATTN$(enc_e, enc_e, enc_e, W_l^{sAtE})$
  $enc_e \leftarrow enc_e + W_l^{ffES}$**max**$(W_l^{ffIE}$**norm**$(enc_e))$
  $enc_s \leftarrow$ ATTN$(enc_s, enc_s, enc_s, W_l^{sAtS})$
  $enc_s \leftarrow enc_s + W_l^{ffOS}$**max**$(W_l^{ffIS}$**norm**$(enc_s))$
  $enc_{es} \leftarrow$ ATTN$(enc_{es}, enc_{es}, enc_{es}, W_l^{sAtES})$
  $enc_{es} \leftarrow enc_{es} + W_l^{ffOES}$**max**$(W_l^{ffIES}$**norm**$(enc_{es}))$
end for
return $[enc_e, enc_s, enc_{es}]$

Algorithm 2 Decoder Pass

Input: enc, encoding of the input sequences of events and alarms, $\hat{X}_a \in V^*$, sequence of alarms predicted until step t
Output: $\hat{P} \in (1,2)^{|V| \times l_a}$, probabilities of alarm tokens given the input and alarms predicted until timestamp t
Hyperparameters: $l_a, d_e, d_f, n_d \in N$, lengths of input sequences of event tokens and signals, and output token sequences for an alarm; embedding depth and depth of the feed-forward layers; number of encoder and decoder layers
Parameters:
$W_e^a \in R^{d_e \times |V|}$, $W_{pe}^a \in R^{d_e \times l_e}$, embedding and positional encoding matrices for alarm tokens
For $l \in [n_d]$: $W_l^{sAt} \in R^{\times}$, $W_l^{cAt} \in R^{\times}$, $W_l^{ffI} \in R^{d_f \times d_e}$, $W_l^{ffO} \in R^{d_e \times d_f}$, parameters of the decoder self- *(sAt)* and cross-attention *(cAt)*, and the inner *(ffI)* and outer *(ffO)* feed-forward layers

Decoding:
for $t \in l_a$ **do**
  $dec_t \leftarrow W_e^a[:\hat{X}_a[t]]\hat{X}_a + W_{pe}^a[:t]\hat{X}_a$
end for
for $l \in [n_d]$ **do**
  $dec \leftarrow$ ATTN$(dec, dec, dec, W_l^{sAt})$
  $enc_s \leftarrow$ ATTN$(dec, enc, enc, W_l^{cAt})$
  $dec \leftarrow dec + W_l^{ffO}$**max**$(W_l^{ffI}$**norm**$(dec))$
end for
return softmax$(W^{aT} dec)$

Algorithm 3 Training Algorithm

Input: $model$, the encoder-decoder Transformer model to be trained, $X_e, X_a \in V^*$, sequences of token IDs of input events and ground truth alarm tokens; $X_s^t \in R^*$, input sequences of signals; L, the loss function; sc, stopping criteria

Training:
Randomly initialize the weights of the $model$
while not sc **do**
  $loss \leftarrow L(X_a, model(X_e, X_s))$
  Calculate gradients of $model$ weights w. r. t $loss$ and update the weights
end while

the predictions. Then, in a backpropagation step, the derivative of the loss function is calculated with respect to the model layers and the weights are updated. The process is repeated over several epochs until convergence. An outline of the procedure can be found in Alg. 3. The exact training parameters are provided in Section 4.3.

4. Experimental evaluation

For validation of APT, its performance is compared on the alarm forecasting task against unimodal Transformers and Transformers with traditional fusion techniques, which are state-of-the-art in academic research. In addition, a late-fusion LSTM-based model (which is state-of-the-art in the industry) has been built and fine-tuned to use as a baseline. Experiments on alarm forecasting were performed using two real datasets and supplemented with an experiment on the operation forecasting task on a simulated dataset. In addition, the effects of encoding alarms at various levels of granularity are explored.

4.1. Datasets

We use two proprietary real-world industrial datasets from two waste incineration plants and one simulated dataset which is used to predict upcoming operations in a chemical batch process to validate APT.

4.1.1. Waste incineration plants datasets

Municipal solid waste incineration is a traditional method of urban waste disposal and simultaneously power generation: the heat released from the incineration of the municipal solid wastes can be used as the input energy to thermal power plants (Yazdani, Salimipour, & Moghaddam, 2020). Fig. 8 shows an example of a waste incineration plant. Waste is burned on a moving grate in the combustion chamber using flue gas and pre-heated air and the resulting steam is used within a cogeneration system to produce energy and heat. The electric energy is produced by a turbine connected to a generator (Moya, Aldás, López, & Kaparaju, 2017; Pavlas, Touš, Klimek, & Bébar, 2011; Xia, Shan, Chen, Du, Huang, & Bai, 2020). Waste-to-Energy technologies such as waste incineration reduce the environmental impact of waste management and at the same time decrease the dependency on fossil fuels (Psomopoulos, Bourka, & Themelis, 2009).

Municipal solid waste consists of multiple heterogeneous fractions with very different physical and chemical properties, and, hence, different combustion and transport behaviors, which makes the waste incineration power generation technology unstable and introduces a lot of uncertainties of the measurement (Wissing, Wirtz, & Scherer, 2017; Ye, Shi, Shi, Zhang, & Zhang, 2021). Together with the high complexity of the system as a whole, this makes the process relatively unstable and generates a fair number of alarms. Early alarm prediction can give the operator more time to react and introduce corrective actions to eliminate possible failures, downtime and negative impact on human

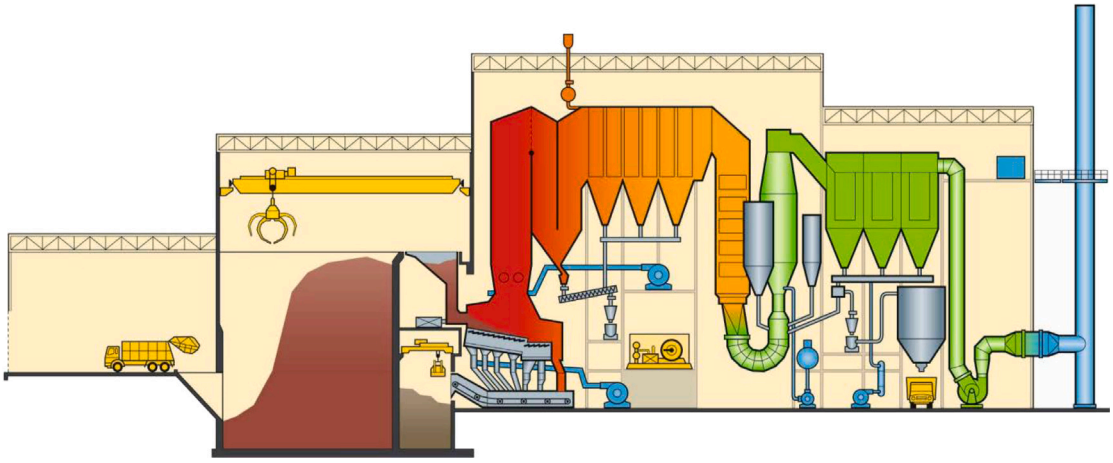


Fig. 8. An example of a waste incineration plant scheme (courtesy of Doosan Lentjes GmbH).

safety and the environment. We propose to use historic data to forecast alarms in a data-driven fashion.

To validate APT on the task of early alarm prediction based on the combination of recent events and signal data, real data from two incineration plants is used, recorded over 6 months. For reasons explained above, the data is characterized by a high degree of noise due to the inherent randomness in the process. In both datasets, the signal data includes 31 continuous process variables selected by a domain expert and resampled at a frequency of 30 seconds (such as flow rates and temperatures of primary air, natural gas and flue gas, amount of incinerated waste, or feed water temperatures and pressures). The event log consists of automatically generated entries indicating state changes throughout the plant, covering different functional areas. These entries are composed of various attributes such as timestamp, tag (a code specifying the component), status, event type, priority, message, etc. Overall, there are 8284205 events logged over the given period in one dataset and 9193762 in the other.

In practice, after an alarm is fired, the changes causing it cascade further and trigger other alarms, potentially leading to alarm floods (IEC 62682 Management of Alarm Systems for the Process Industries, 2014). In addition, unless properly handled, an alarm can deteriorate into higher severity if the next critical threshold is crossed. For this reason, for the sake of this task, only the first alarm is predicted, since otherwise the operator would already be aware of the anomalous situation, even though predicting subsequent alarms could be easier due to the alarms already recorded in the log as well as a more prominent footprint of the anomaly in the signal data. Therefore, the model is trained on time windows preceding alarms such that there are no other alarms in the same functional area 20 min before, balanced with an equal number of time windows not followed by alarms. Specifically, for every sample, a window is taken starting 20 min before an alarm and ending within a forecasting horizon of 5 min before the alarm.

Overall, there are 10,714 samples (half of them positive, half alarm-free) in dataset A, which is randomly split into train, dev and test sets of 8572, 1071 and 1071 samples, respectively (0.80/0.10/0.10). There are 102 different alarm types, 98 of which are seen at least once in the train set. In dataset B, there are 33,534 samples, randomly split into train, dev and test sets of 26828, 3353, 3355 samples, respectively, with 194 different alarm types, 188 of which are at least once seen in the train set. With this, the datasets are balanced w. r. t. binary prediction and highly unbalanced w. r. t. multiclass prediction, with half of the samples labeled as having `No_Tag` and the rest being distributed as seen in Figs. 9(a) and 9(b). To ensure fair comparison of different model architectures we therefore ensure that the samples are split into train, dev and test sets in a balanced way and fix the split across all experiments.

Initially, various data augmentation techniques have also been explored, such as adding noise to signal data as well as oversampling events based on temporal or topological proximity to the alarm or in a random fashion, yet these did not yield any significant advantage. Given that this work is focused on comparing multimodal learning techniques, for the final experiments, the data was used as is, without augmentation.

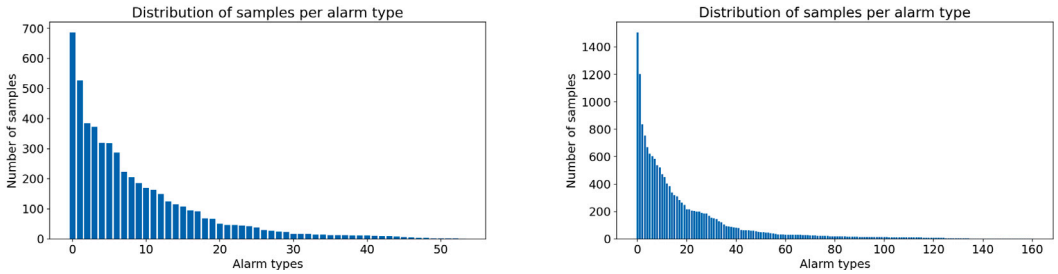
Depending on the decoding option (characters, 'morphemes', or entire tags — see Section 3.5), at each step the model has to choose one of 43, 103, or 108 classes, respectively, in dataset A and one of 43, 166, or 200 classes in dataset B (including auxiliary tokens like `Alarm` or `No_alarms`.).

To sum up, both datasets pose several challenges:

1. The datasets are relatively small, considering the number of classes and the number of samples per class, as well as the complexity of the task;
2. The distribution of classes is highly unbalanced, with the vast majority of classes represented by very few samples;
3. The input events are very sparse: with data coming from the entire plant, events are spread across numerous plant functions, equipment and components;
4. The modalities cannot be directly aligned;
5. The data comes from real plants, characterized by a high degree of randomness in the process, and contains a lot of noise;
6. There are a lot of gaps with individual process variables missing. Moreover, a lot of event data is lost due to trimming;
7. Events are logged across the plants and domain expertise would be required to identify which events could be the cause of specific alarms. It is also likely that relevant events get trimmed and do not appear in the input;
8. Early prediction of alarms means that the underlying changes causing the alarm might not have yet manifested themselves in the data;
9. The available process variables are not linked to alarm tags. The task could be easier if all signals corresponding to different alarm tags could be used: in that case, given enough data, the model could learn the thresholds. However, a mapping from alarm tags to appropriate process variables is unavailable, and only 31 signals are used to predict 102 or 194 alarm types.

4.1.2. Simulated dataset

In addition to real datasets, our approach is validated on a more tractable task of forecasting operations in a chemical batch process using a simulated dataset (Tan, 2022). Batch production processes consist of cyclic sequences of operations like heating, cooling, chemical



(a) Real data (plant A) (b) Real data (plant B)

Fig. 9. Alarm type distribution. Alarm types in both datasets are highly unbalanced with ca. 20% of the alarm types accounting for 90% of the samples.

reactions, or stirring. The specific instances of such operations can vary considerably even for the same product (Just, Khaydarov, Klöpper, Bähner, & Urbas, 2022), e.g. due to variation in the raw material. A reliable prediction when such operation will end are valuable inputs for production, logistics and personnel planning. The dataset covers the simulated equivalent of more than 2 months of a batch production process and contains 4268 events of 18 classes designating the start of a new operation (such as filling, processing, draining and cleaning) and signal data including 5 continuous process variables resampled at a frequency of 15 seconds (vessel filling levels, motor rotation speed, cooling water flow rate, as well as steam flow).

The model is trained on time windows preceding each new operation starting 36 min before it begins and ending within a forecasting horizon of 6 min. These windows are balanced with an equal number of time windows not followed by new operations within the forecasting horizon. The task solved here is to first make a binary prediction of whether a new operation is going to begin after the forecasting horizon and, if yes, which operation it would be.

Overall, the simulated dataset contains 5496 samples, which are randomly split into train, dev and test sets of 4398, 549 and 549 samples, respectively (0.80/0.10/0.10). The events in this dataset belong to 25 classes (including auxiliary tokens) and refer to discrete operations and not hierarchical tag IDs, therefore they are not split into characters or 'morphemes' and each is treated as a separate class (cf. entire tag in the real datasets).

4.2. Baseline models

As a multimodal Transformer-based model with hybrid fusion, APT is compared against several baselines: a multimodal LSTM model (which is state-of-the-art in the industry), as well as unimodal Transformers and Transformers with traditional fusion techniques, which are state-of-the-art in academic research, yet have not been applied to solve industrial tasks before. Specifically, the following baseline models are used:

1. A multimodal LSTM-based model, in which both input modalities are passed through LSTM encoders separately and concatenated into a joint hidden representation, which is further passed to the decoder.
2. A Transformer-based model relying solely on signal data as input.
3. A Transformer-based model using only events.
4. Three Multimodal Transformer-based models with 'skip', early and late fusion, respectively.

For more details on different fusion types, we refer the reader to Section 3.4.

4.3. Training parameters and scoring

The models are trained using Adam optimizer (Kingma & Ba, 2014) with $\beta_1 = 0.9$, $\beta_2 = 0.99$ and a learning rate of 1.0^{-3}. The dropout rate is 0.3. The batch size is 128. The model is trained using a cross-entropy loss averaged over all tokens (excluding padding), which, depending on alarm encoding, can be represented by a complete alarm tag, its parts ('morphemes') or individual characters.

Due to the highly unbalanced distribution of alarm classes, the main evaluation metric used is F1 score, computed separately for binary and multiclass predictions. To calculate F1 score for binary predictions, only the first token of each sample is considered (`Alarm/No_alarms`). For multiclass predictions, the subsequent tokens of each sample are concatenated to bring them back to the initial form in case they were encoded as characters or 'morphemes'. The F1 score is calculated on these concatenated tokens (e.g., `XOHNA7OFQO13XH52`).

In Table 1, one can see examples of correct and incorrect predictions across different alarm encoding methods. The rows marked as 'True' show the ground truth, the rows marked as 'Pred' contain predictions with differences highlighted in accordance with the alarm encoding method: during training, a predicted token that differs from the truth is considered wrong, regardless of the Levenstein distance between the true and predicted alarm tag on character level. At test time, tokens constituting one alarm tag are concatenated, and even if one of them is wrong, the whole tag is considered as incorrect for evaluation purposes.

Since the multiclass F1 is strict and ignores partial matches in alarm tag predictions, the evaluation also includes the BLEU score (Papineni, Roukos, Ward, & Zhu, 2002) to compare representation capacity of APT against the baseline models. The BLEU score is a precision-oriented metric traditionally used for assessing natural language generation tasks. The BLEU score essentially calculates the proportion of words in a generated sentence which are also found in a reference sentence. However, in this work the score is only applicable for comparison of fusion methods but not different alarm encodings: BLEU is highly sensitive to sentence length, therefore it is not comparable at different levels of granularity.

4.4. Results and discussion

The results for real datasets are summarized in Tables 2 (with alarms encoded as whole tags), 3 (with alarms encoded as 'morphemes'), and 4 (with alarms encoded as characters). The tables present average F1 scores for binary (`Alarm/No_alarms`) and multiclass predictions (specific tags) for two plants across different models.

The results for the simulated dataset are summarized in Table 5 (operations are encoded as whole tags for all of the experiments). In addition, Fig. 10 shows the confusion matrices for the simulated dataset.

Overall, the scores are highest for the simulated dataset, which is small but has much less noise and fewer classes, and lowest for the real dataset from plant A, which is both highly noisy and relatively small. This is in line with findings of a comparative study of multimodal

Table 1
Examples of correct and incorrect predictions for different alarm encoding strategies. During training, with alarms encoded as characters, the model is penalized only for incorrectly predicted characters; with alarms encoded as 'morphemes', the loss considers incorrect 'morphemes'; with alarms encoded as classes, the whole tag must match to be correct. During test time, the evaluation is done at the level of whole tags.

		Classes	'Morphemes'	Characters
Correct	True	Alarm, X0EKG11CP001ZH52	Alarm, X0EKG, 11, CP001, ZH52	Alarm, X, 0, H, N, A, 7, 0, F, Q, 0, 1, 3, X, H, 5, 2
	Pred	Alarm, X0EKG11CP001ZH52	Alarm, X0EKG, 11, CP001, ZH52	Alarm, X, 0, H, N, A, 7, 0, F, Q, 0, 1, 3, X, H, 5, 2
Incorrect	True	Alarm, X0SAB05CP001XH52	Alarm, X0SAB, 05, CP001, XH52	Alarm, X, 0, H, N, A, 7, 0, F, Q, 0, 1, 3, X, H, 5, 2
	Pred	Alarm, X0LBN91CT901XH52	Alarm, X0HNA, 70, FQ013,XH52	Alarm, X, 0, H, B, K, 4,0, F, Q, 9, 0, 1,X, H, 5, 2

Table 2
Effects of fusion on prediction accuracy on the real datasets with labels encoded as classes. For F1 Bin (binary predictions), F1 MC (multiclass) and BLEU, higher is better. **Bold**: best, underlined: second best. APT is marked with an asterisk (*) if better than either early or late fusion and with two (**) if better than both.

	Plant A			Plant B		
Fusion Type	F1 Bin	F1 MC	BLEU	F1 Bin	F1 MC	BLEU
LSTM	58.9 ±2.5	34.3 ±0.9	53.1 ±4.4	84.7 ±1.0	44.1 ±1.4	65.1 ±0.7
Signal Only	58.8 ±0.6	47.7 ±0.1	41.2 ±6.5	85.2 ±0.5	50.9 ±0.1	55.2 ±1.7
Events Only	68.6 ±1.4	47.5 ±0.0	52.1 ±1.9	82.5 ±0.4	50.3 ±0.0	62.2 ±0.6
Skip Fusion	**69.8 ±0.7**	44.6 ±7.4	51.1 ±2.3	84.0 ±0.3	50.6 ±0.2	62.3 ±1.1
Early Fusion	68.2 ±1.0	44.6 ±7.3	52.5 ±1.1	85.1 ±1.1	51.1 ±0.5	64.3 ±2.2
Late Fusion	69.0 ±0.9	**48.2 ±0.2**	**53.8 ±1.4**	84.9 ±1.8	51.0 ±0.4	65.1 ±1.9
APT	69.1 ±1.3**	45.2 ±7.1*	53.1 ±1.1*	**86.2 ±1.8****	**51.5 ±0.2****	**66.1 ±2.8****

Table 3
Effects of fusion on prediction accuracy on the real datasets with labels encoded as 'morphemes'. For F1 Bin (binary predictions), F1 MC (multiclass) and BLEU, higher is better. **Bold**: best, underlined: second best. APT is marked with an asterisk (*) if better than either early or late fusion and with two (**) if better than both.

	Plant A			Plant B		
Fusion Type	F1 Bin	F1 MC	BLEU	F1 Bin	F1 MC	BLEU
Signal Only	59.1 ±1.5	43.4 ±5.5	33.2 ±0.9	85.5 ±0.5	45.3 ±1.9	39.7 ±0.9
Events Only	68.3 ±1.1	**50.7 ±5.0**	39.4 ±2.8	82.6 ±0.7	50.4 ±1.4	44.0 ±0.6
Skip Fusion	**69.8 ±0.8**	48.6 ±4.5	40.4 ±0.9	83.9 ±0.8	50.4 ±0.6	44.1 ±0.9
Early Fusion	68.8 ±0.8	48.1 ±4.8	40.5 ±1.7	86.4 ±1.7	53.6 ±1.1	48.1 ±1.3
Late Fusion	**69.8 ±1.1**	48.7 ±4.1	40.4 ±1.3	85.7 ±1.7	52.9 ±1.9	47.0 ±2.3
APT	69.1 ±1.9*	50.0 ±1.7**	**41.0 ±2.2****	**86.9 ±1.4****	**54.5 ±1.4****	**48.7 ±1.1****

Table 4
Effects of fusion on prediction accuracy on the real datasets with labels encoded as characters. For F1 Bin (binary predictions), F1 MC (multiclass) and BLEU, higher is better. **Bold**: best, underlined: second best. APT is marked with an asterisk (*) if better than either early or late fusion and with two (**) if better than both.

	Plant A			Plant B		
Fusion Type	F1 Bin	F1 MC	BLEU	F1 Bin	F1 MC	BLEU
Signal Only	58.1 ±0.9	41.3 ±8.4	42.9 ±9.8	85.5 ±0.5	45.4 ±2.4	51.0 ±1.3
Events Only	68.5 ±0.5	47.2 ±3.8	49.9 ±5.7	81.9 ±0.8	50.8 ±1.6	55.0 ±1.2
Skip Fusion	69.5 ±0.6	47.7 ±3.6	50.8 ±6.0	83.0 ±1.3	51.3 ±1.1	55.7 ±0.9
Early Fusion	68.4 ±1.2	47.8 ±3.2	49.4 ±3.6	86.4 ±2.1	54.1 ±2.4	60.0 ±2.3
Late Fusion	**70.7 ±0.8**	**50.1 ±3.5**	47.2 ±4.4	86.0 ±1.9	**54.4 ±2.7**	58.2 ±2.6
APT	69.2 ±1.4*	47.9 ±2.7*	**51.2 ±0.5****	**86.8 ±0.9****	54.2 ±1.3*	**60.4 ±1.3****

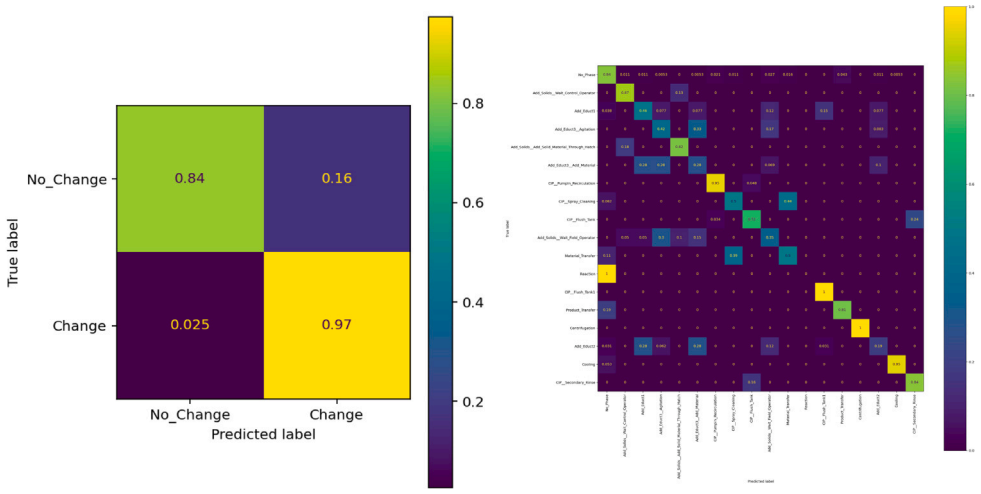
Transformers (Hendricks, Mellor, Schneider, Alayrac, & Nematzadeh, 2021), which showed that the quality of pretraining datasets is more significant than their size.

As can be seen, APT performs either on par with or better than other fusion methods, unimodal Transformers and LSTM for both binary and multiclass predictions. The effect is even stronger for the BLEU score, which would have been used to evaluate a model's performance in the domain of natural language and which is used here as a proxy for representation learning capacity of the models. In Tables 2, 3, 4 and 5, it can be seen that APT is almost always best (bold) or second best (underlined). Additionally, when compared to multimodal Transformers with traditional early and late fusion, it outperforms either one of them (marked with an asterisk) or both (marked with two asterisks). This suggests that, while, depending on the dataset and the number of classes, either early or late fusion may perform better than the other, in most cases with APT, which relies on the novel hybrid fusion, one gains advantage either over both or at least one.

Table 5
Effects of fusion on prediction accuracy on the simulated dataset with labels encoded as classes. For F1 Bin (binary predictions), F1 MC (multiclass) and BLEU, higher is better. **Bold**: best, underlined: second best. APT is marked with an asterisk (*) if better than either early or late fusion and with two (**) if better than both.

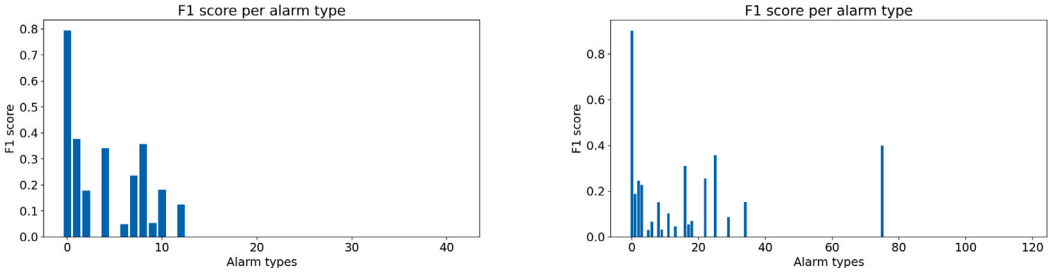
	F1 Bin	F1 MC	BLEU
LSTM	87.6 ±1.1	66.5 ±6.9	78.7 ±2.7
Signal Only	94.4 ±0.2	67.6 ±1.0	78.0 ±1.1
Events Only	89.8 ±0.3	45.9 ±0.4	70.4 ±0.6
Skip Fusion	**94.8 ±0.3**	68.5 ±1.5	79.2 ±0.4
Early Fusion	94.7 ±0.4	69.0 ±0.8	79.2 ±0.6
Late Fusion	94.5 ±0.1	68.9 ±0.7	**79.5 ±1.3**
APT	**94.8 ±0.1****	**69.1 ±1.0****	79.3 ±0.9*

On the other hand, preliminary experiments produced comparatively high scores of models with 'skip' fusion, which suggests that



(a) Binary Predictions (b) Multiclass Predictions

Fig. 10. Confusion matrices for operation predictions with APT on the simulated dataset.



(a) Real Data (Plant A) (b) Real Data (Plant B)

Fig. 11. F1 score distribution per alarm type sorted by alarm type frequency (test set).

residual connections including 'raw' inputs that have not been passed through a Transformer block need to be further investigated.

With regard to alarm tag encoding, one can see a benefit of using alarms encoded as 'morphemes' or characters instead of whole tags. The binary F1 scores (for the prediction of the presence or absence of an alarm) are essentially the same across the three settings, which makes sense because the granular alarm encoding affects only the alarm tags. BLEU scores are used for the comparison of fusion techniques but are inapplicable for comparing alarm encodings since the vocabulary sizes for alarm tags encoded as characters, morphemes or whole words are different. Multiclass F1 scores however, reflecting how accurately alarm tags are predicted, show a clear advantage of the granular encoding, 3 to 5 percentage points for morphemes over the entire tags. It should also be observed that, although forecasting a specific alarm tag remains hard, the model easily learns its structure: tag parts corresponding to plant, function, equipment and component are always predicted in proper order (see examples in Table 1). This also supports our intuition that sequential prediction of alarm tags can leverage the model's representation learning capacity and yield better results over simple one-step multiclass classification.

As described above, the task at hand is complex due to multiple factors such as sparse and noisy input data, a high number of classes (108 and 200 classes in real datasets — cf. 25 classes in the simulated dataset) and the highly unbalanced distribution of labels (as illustrated in Figs. 9(a) and 9(b)). Considering the distribution of average F1 scores sorted by alarm type frequency (see Figs. 11(a) and 11(b)), the complexity of the task is caused not only by the highly skewed distribution of labels, but also by that of events and signal data in the input (see the outline of data limitations in Section 4.1.1). Nonetheless, the experiments demonstrate competitive performance of our model.

5. Conclusion

We introduced APT, a multimodal Transformer-based model for the industrial domain, where, to the best of our knowledge, multimodal Transformers have not yet been used. The Transformer has been chosen as the state-of-the-art architecture allowing to efficiently capture long-distance dependencies in sequences. We used APT to tackle a complex real-world task of early prediction of alarms belonging to over a hundred of classes with a highly unbalanced distribution. APT predicts, based on the combination of recent events and signal data as input, whether an alarm is going to be triggered after the given forecasting horizon and, if yes, it also predicts an alarm type. In a series of experiments, our model outperformed unimodal baselines and an LSTM-based multimodal baseline. We proposed a new hybrid fusion method to combine modalities and compared it to early and late fusion strategies. The experiments show that the proposed fusion method achieves comparable or better results. The advantage of using hybrid fusion is thus that it allows to replace early or late fusion methods and combine the benefits of both.

In addition, relying on domain knowledge, we introduced several methods of alarm encoding at different levels of granularity and demonstrated the benefits of the granular approach whereby alarm tag labels are predicted sequentially as compared to prediction of the whole tag.

For practical purposes, our approach allows improving the results by postprocessing model outputs, such as changing the prediction of a tag

based on more reliable binary outputs and searching the closest valid tag in case the tag predicted at the level of characters or morphemes does not happen to be in the set of valid tags, but this is out of scope of the present work.

CRediT authorship contribution statement

Nika Strem: Conceptualization, Methodology, Data curation, Writing – original draft. **Devendra Singh Dhami:** Conceptualization, Writing – review & editing. **Benedikt Schmidt:** Resources, Data curation. **Benjamin Klöpper:** Conceptualization, Resources. **Kristian Kersting:** Conceptualization, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Acknowledgments

This work was supported by ABB AG Forschungszentrum Ladenburg, the Collaboration Lab 'AI in Construction' (AICO), the ICT-48 Network of AI Research Excellence Center 'TAILOR' (EU Horizon 2020, GA No 952215) and the HMWK cluster project 'The Third Wave of AI'.

Data availability

The data that has been used is confidential.

References

Akbari, H., Yuan, L., Qian, R., Chuang, W.-H., Chang, S.-F., Cui, Y., et al. (2021). Vatt: Transformers for multimodal self-supervised learning from raw video, audio and text. *Advances in Neural Information Processing Systems*, *34*.

Boussioux, L., Zeng, C., Guénais, T., & Bertsimas, D. (2022). Hurricane forecasting: A novel multimodal machine learning framework. *Weather and Forecasting*.

Cai, S., Palazoglu, A., Zhang, L., & Hu, J. (2019). Process alarm prediction using deep learning and word embedding methods. *ISA Transactions*, *85*, 274–283.

Chatterjee, J., & Dethlefs, N. (2020). A dual transformer model for intelligent decision support for maintenance of wind turbines. In *2020 international joint conference on neural networks* (pp. 1–10).

Chen, S., Guhur, P.-L., Schmid, C., & Laptev, I. (2021). History aware multimodal transformer for vision-and-language navigation. *Advances in Neural Information Processing Systems*, *34*.

Cho, J., Lei, J., Tan, H., & Bansal, M. (2021). Unifying vision-and-language tasks via text generation. URL http://arxiv.org/abs/2102.02779.

Datong, L., Yu, P., & Xiyuan, P. (2009-05). Fault prediction based on time series with online combined kernel SVR methods. In *2009 IEEE instrumentation and measurement technology conference* (pp. 1163–1166).

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding.

Dhami, D. S., Yan, S., Kunapuli, G., Page, D., & Natarajan, S. (2021). Predicting drug-drug interactions from heterogeneous data: An embedding approach. In *International conference on artificial intelligence in medicine* (pp. 252–257). Springer.

Di Lello, E., Klotzbucher, M., De Laet, T., & Bruyninckx, H. (2013). Bayesian time-series models for continuous fault detection and recognition in industrial robotic tasks. In *2013 IEEE/RSJ international conference on intelligent robots and systems* (pp. 5827–5833). IEEE.

Essen, V. K. G. (2021). KKS Kraftwerk-Kennzeichensystem.

Fadzail, N. F., Mat Zali, S., Mid, E. C., & Jailani, R. (2022). Application of automated machine learning (AutoML) method in wind turbine fault detection. *Journal of Physics: Conference Series*, *2312*(1), Article 012074.

Giurgiu, I., & Schumann, A. (2019). Explainable failure predictions with RNN Classifiers based on time series data. arXiv:1901.08554.

Helbing, G., & Ritter, M. (2018-12-01). Deep learning for fault detection in wind turbines. *Renewable and Sustainable Energy Reviews*, *98*, 189–198.

Hendricks, L. A., Mellor, J., Schneider, R., Alayrac, J.-B., & Nematzadeh, A. (2021). Decoupling the role of data, attention, and losses in multimodal transformers.

Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J., et al. (2001). Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.

Hu, R., & Singh, A. (2021). UniT: Multimodal multitask learning with a unified transformer. URL http://arxiv.org/abs/2102.10772.

IEC 62682 Management of Alarm Systems for the Process Industries (2014). *Standard No. IEC 62682*, IEC, Geneva, Switzerland: International Electrotechnical Commission.

Inceoglu, A., Aksoy, E. E., Cihan Ak, A., & Sariel, S. (2021). FINO-net: a deep multimodal sensor fusion framework for manipulation failure detection. In *2021 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 6841–6847).

Just, G., Khaydarov, V., Klöpper, B., Bähner, F. D., & Urbas, L. (2022). Hidden Markov Models und Active Learning zur automatisierten Kennzeichnung von Batchphasen in der Prozessindustrie. *Automation*.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization.

Klein, G., Kim, Y., Deng, Y., Senellart, J., & Rush, A. M. (2017). Opennmt: Open-source toolkit for neural machine translation.

Koltsidopoulos Papatzimos, A., Thies, P. R., & Dawood, T. (2019). Offshore wind turbine fault alarm prediction. *Wind Energy*, *22*(12), 1779–1788.

Langone, R., Alzate, C., Bey-Temsamani, A., & Suykens, J. A. K. (2014). Alarm prediction in industrial machines using autoregressive LS-svm models. In *2014 IEEE symposium on computational intelligence and data mining (CIDM)* (pp. 359–364).

Langone, R., Cuzzocrea, A., & Skantzos, N. (2020-11-01). Interpretable Anomaly Prediction: Predicting anomalous behavior in industry 4.0 settings via regularized logistic regression tools. *Data & Knowledge Engineering*, *130*, Article 101850.

Li, H., Huang, J., Huang, J., Chai, S., Zhao, L., & Xia, Y. (2021). Deep multimodal learning and fusion based intelligent fault diagnosis approach. *Journal of Beijing Institute of Technology*, *30*(2), 172–185.

Li, H., Qian, B., Parikh, D., & Hampapur, A. (2013-10). Alarm prediction in large-scale sensor networks. A case study in railroad. In *2013 IEEE international conference on big data* (pp. 7–14). IEEE.

Li, Z., Wang, Y., & Wang, K.-S. (2017). Intelligent predictive maintenance for fault diagnosis and prognosis in machine centers: Industry 4.0 scenario. *Advances in Manufacturing*, *5*(4), 377–387.

Limoyo, O., Ablett, T., & Kelly, J. (2022). Learning sequential latent variable models from multimodal time series data. URL http://arxiv.org/abs/2204.10419.

Liu, H., Fang, T., Zhou, T., Wang, Y., & Wang, L. (2018). Deep learning-based multimodal control interface for human-robot collaboration. *Procedia CIRP*, *72*, 3–8.

Lomov, I., Lyubimov, M., Makarov, I., & Zhukov, L. E. (2021). Fault detection in Tennessee Eastman process with temporal deep learning models. *Journal of Industrial Information Integration*, *23*, Article 100216.

Lu, J., Batra, D., Parikh, D., & Lee, S. (2019). ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Vol. 32*, In *Advances in neural information processing systems*. Curran Associates, Inc., URL https://proceedings.neurips.cc/paper/2019/hash/c74d97b01eae257e44aa9d5bade97baf-Abstract.html.

Lucke, M., Stief, A., Chioua, M., Ottewill, J. R., & Thornhill, N. F. (2020-01-01). Fault detection and identification combining process measurements and statistical alarms. *Control Engineering Practice*, *94*, Article 104195.

Moya, D., Aldás, C., López, G., & Kaparaju, P. (2017). Municipal solid waste as a valuable renewable energy resource: A worldwide opportunity of energy recovery by using Waste-To-Energy Technologies. *Energy Procedia*, *134*, 286–295.

Nambiar, A., Heflin, M., Liu, S., Maslov, S., Hopkins, M., & Ritz, A. (2020). Transforming the language of life: Transformer neural networks for protein prediction tasks. In *Proceedings of the 11th ACM international conference on bioinformatics, computational biology and health informatics* (pp. 1–8).

Noever, D., Ciolino, M., & Kalin, J. (2020). The chess transformer: Mastering play using generative language models.

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics* (pp. 311–318). Association for Computational Linguistics.

Pavlas, M., Touš, M., Klimek, P., & Bébar, L. (2011). Waste incineration with production of clean and reliable energy. *Clean Technologies and Environmental Policy*, *13*(4), 595–605.

Pramanik, S., Agrawal, P., & Hussain, A. (2019). Omninet: A unified architecture for multi-modal multi-task learning.

Proto, S., Ventura, F., Apiletti, D., Cerquitelli, T., Baralis, E., Macii, E., et al. (2019-07). PREMISES, a scalable data-driven service to predict alarms in slowly-degrading multi-cycle industrial processes. In *2019 IEEE international congress on big data (bigDataCongress)* (pp. 139–143). IEEE.

Psomopoulos, C. S., Bourka, A., & Themelis, N. J. (2009). Waste-to-energy: A review of the status and benefits in USA. *Waste Management*, *29*(5), 1718–1724.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning* (pp. 8748–8763). PMLR.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, *1*(8), 9.

Rahman, W., Hasan, M. K., Lee, S., Zadeh, A., Mao, C., Morency, L.-P., et al. (2020). Integrating multimodal information in large pretrained transformers. *2020*, In *Proceedings of the conference. association for computational linguistics. meeting* (p. 2359). NIH Public Access.

Reinartz, C., Kulahci, M., & Ravn, O. (2021). An extended Tennessee Eastman simulation dataset for fault-detection and decision support systems. *Computers & Chemical Engineering*, *149*, Article 107281.

Stauffer, T., & Clarke, P. (2016). Using alarms as a layer of protection. *Process Safety Progress*, *35*(1), 76–83.

Sun, C., Baradel, F., Murphy, K., & Schmid, C. (2019). Learning video representations using contrastive bidirectional transformer. URL http://arxiv.org/abs/1906.05743.

Tan, R. (2022). Datasets from multiple cycles. http://dx.doi.org/10.57826/KEEN/ODU6MA.

Tan, H., & Bansal, M. (2019). Lxmert: Learning cross-modality encoder representations from transformers.

Tiulpin, A., Klein, S., Bierma-Zeinstra, S., Thevenot, J., Rahtu, E., Meurs, J. v., et al. (2019). Multimodal machine learning-based knee osteoarthritis progression prediction from plain radiographs and clinical data. *Scientific Reports*, *9*(1), 1–11.

Tsai, Y.-H. H., Bai, S., Liang, P. P., Kolter, J. Z., Morency, L.-P., & Salakhutdinov, R. (2019). Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 6558–6569). Association for Computational Linguistics, URL https://aclanthology.org/P19-1656.

Vaswani, A., Bengio, S., Brevdo, E., Chollet, F., Gomez, A. N., Gouws, S., et al. (2018). Tensor2tensor for neural machine translation.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30*.

Villalobos, K., Suykens, J., & Illarramendi, A. (2021). A flexible alarm prediction system for smart manufacturing scenarios following a forecaster–analyzer approach. *Journal of Intelligent Manufacturing*, *32*(5), 1323–1344.

Wang, J., Hu, X., Gan, Z., Yang, Z., Dai, X., Liu, Z., et al. (2021). UFO: A UniFied TransfOrmer for vision-language representation learning. URL http://arxiv.org/abs/2111.10023.

Wang, X., & Liang, D. (2020). LSTM-based alarm prediction in the mobile communication network. In *2020 IEEE 6th international conference on computer and communications* (pp. 561–567).

Wang, P., Yang, A., Men, R., Lin, J., Bai, S., Li, Z., et al. (2022). Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. URL http://arxiv.org/abs/2202.03052.

Wissing, F., Wirtz, S., & Scherer, V. (2017). Simulating municipal solid waste incineration with a DEM/CFD method–Influences of waste properties, grate and furnace design. *Fuel*, *206*, 638–656.

Xia, Z., Shan, P., Chen, C., Du, H., Huang, J., & Bai, L. (2020). A two-fluid model simulation of an industrial moving grate waste incinerator. *Waste Management*, *104*, 183–191.

Yang, Z., Baraldi, P., & Zio, E. (2021). A multi-branch deep neural network model for failure prognostics based on multimodal data. *Journal of Manufacturing Systems*, *59*, 42–50, URL https://linkinghub.elsevier.com/retrieve/pii/S0278612521000078.

Yazdani, S., Salimipour, E., & Moghaddam, M. S. (2020). A comparison between a natural gas power plant and a municipal solid waste incineration power plant based on an emergy analysis. *Journal of Cleaner Production*, *274*.

Ye, B., Shi, B., Shi, M., Zhang, L., & Zhang, R. (2021). Process simulation and comprehensive evaluation of a system of coal power plant coupled with waste incineration. *Waste Management & Research*, *39*(6), 828–840.

Zhao, H., Hu, Y., Ai, X., Hu, Y., & Meng, Z. (2018-03). Fault detection of Tennessee Eastman process based on topological features and SVM. *IOP Conference Series: Materials Science and Engineering*, *339*, Article 012039.

Zhu, J., Wang, C., Li, C., Gao, X., & Zhao, J. (2016). Dynamic alarm prediction for critical alarms using a probabilistic model. *Chinese Journal of Chemical Engineering*, *24*(7), 881–885.