Question 1:

The most logical sequence prediction method would be a sequence-to-vector model in the situation of predicting the operating mode of a turbine based on two sensor reading time series. This is due to the fact that we are attempting to predict a single output—the operating mode—from a set of several sequential inputs (the sensor readings). This is a classification task.

A sequence-to-vector framework creates a fixed-size output vector from a sequence of input data. This method is frequently applied to a variety of Ai problems, including sentiment analysis, language modelling, and time series classification.

We can apply a sequence-to-vector model, such as a recurrent neural network (RNN) or a long short-term memory (LSTM) network, to the problem of predicting the operating mode of a turbine. These models generate a fixed-size output vector (in our example, the anticipated operating mode) from a series of input data (in our case, the two sensor reading time series). The RNN or LSTM network can be trained to recognise the temporal correlations between the input time series data and map them to the appropriate operating mode for the output.

A 3-dimensional tensor of shape (batch_size, time_steps, features) would be the data shape required for this approach, where batch_size denotes the quantity of samples in each batch, time_steps denotes the length of each time series (the number of readings), and features denotes the quantity of features for each time step (in our case, two sensors). A 2-dimensional tensor of the form (batch_size, num_classes), where num_classes is the total number of operational modes, would be the output shape.

Question 4:

Business applications that analyse time-series data with short-term dependencies can benefit greatly from Conv1D layers. These kinds of applications, which concentrate on local patterns within a sequence, could include anomaly identification, sentiment analysis, and speech recognition. Conv1D layers are also helpful when the input sequence is very long and contains a lot of time steps because recurrent layers may be too expensive computationally in these circumstances.

On the other hand, when dealing with long-term dependencies in the data, recurrent layers like SimpleRNN, LSTM, or GRU are favoured. This is crucial for business systems that use speech recognition, machine translation, and language modelling. Recurrent layers are particularly effective at capturing these dependencies when the aim is to simulate the sequence's long-term structure.

When the time series data shows spatial structure, utilising convolutional neural networks with 2D convolutional layers is quite helpful. This can occur in situations when each time step in the sequence correlates to a specific spatial location in the image, as in applications like image classification. In these situations, transforming the time series into images can aid

in capturing the spatial organisation of the data, enabling the application of convolutional neural networks with 2D convolutional layers. Paper shows how this method may be used to classify faults in wind turbines by combining several time series into a set of images that are then used to train a CNN classifier. It might not be as useful, though, if the temporal connections are intricate and difficult to describe in 2D.

Question 5:

The method shown here, which employs Conv1D layers for time-series data analysis, can be very helpful since it can identify local patterns in the time series data without relying on prior values, which is a crucial component of recurrent neural networks (RNNs). As a result, it is especially suitable for issues where the patterns in the data are subject to rapid change and/or are significantly influenced by short-term causes.

In order to capture local patterns in the data, such as trends or periodicities, Conv1D layers apply convolutional filters to local segments of the time series data. This allows them to do so without requiring to remember previous values. Subsequent layers can then integrate these local patterns to create a global representation of the data.

Conv1D layers are computationally efficient and may be trained more quickly than RNNs, which need sequential processing and often experience memory decay and vanishing gradients. Additionally, Conv1D layers' local approach to pattern identification may be more resistant to noise and better able to detect transient changes in the input.

RNNs still have a place in time-series analysis, especially for issues involving long-term dependencies and situations in which the sequence of occurrences is crucial, such as in speech or natural language processing jobs. The choice of strategy will ultimately depend on the details of the issue at hand and the features of the time-series data being examined.