Department of Computing

Imperial College of Science, Technology and Medicine

H-PERL: The Hybrid Property, Event and Relation Learner

Author:
Ross Irwin

Supervisors:
Prof. Alessandra Russo
Dr. Krysia Broda
Dr. Jorge Lobo

June 8, 2020

Chapter 1

Trained Model

In contrast to the hardcoded model, the trained H-PERL model does not use manually engineered relations or events components, and must therefore rely on using components which can be trained. The trained model also uses the QA-data version of the OceanQA dataset, rather than the full-data version which the hardcoded model was able to use to train its properties component. This means that the trained model needs to rely on the data contained in QA pairs alone to train its components.

As with the hardcoded model, full performance evaluation details of the trained model and some of its components can be found in Chapter **??**.

1.1 Properties

As mentioned in Chapter **??**, property questions in the OceanQA dataset ask the model to find a property value for a specific object. This object, however, can contain a reference to a property value. This means that, in some cases, knowledge of object properties is required in order to find the specified object in the frame. For example, if a question asked "What colour was the upward-facing fish in frame 12?", and there three fish, each with unique rotations, in frame 12, one would need knowledge of object properties in order to select the correct image of the fish. This causes a major problem when collating the training data for the properties component; the model needs a trained property extractor in order to find the images to train the property extractor with.

In this section we propose a solution to overcome this problem which utilises semi-supervised learning in order to label all of the objects in the dataset with property values. Once these labels have been found, the property component can be trained in the same way as the hardcoded model, outlined in Chapter **??**.

The algorithm for finding these labels consists of the following high-level steps, which are described in further detail below:

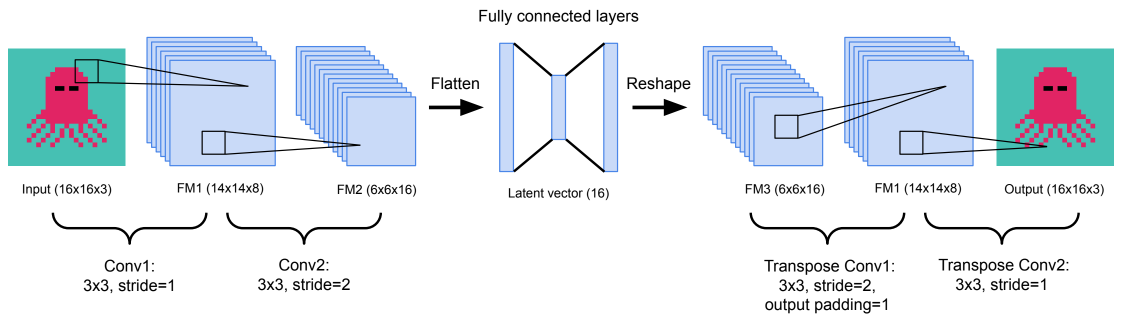

Figure 1.1: An illustration of the autoencoder architecture. Although not shown here, the network contains batch normalisation layers between each pair of convolution layers. FM stands for feature maps, and the dimensions of each feature maps are given as $(height, width, number\ of\ feature\ maps)$.

1. An autoencoder neural network is trained to extract a latent vector from each object image.
2. Each object image in the training data is encoded into a vector using the autoencoder, and, for each object class, the object vectors corresponding to that class are clustered.
3. For each object class, a number of objects in that class

The first step of the algorithm for finding these labels involves training an autoencoder neural network to extract a 16-dimensional latent vector from each object image. This network is trained in an unsupervised manner using a sample of the objects detected by the object detector in the training data, where each object class is equally represented in the sample. The architecture of the network is shown in Figure 1.1. The network is trained with a learning rate of 0.001 for 5 epochs.

1.2 Relations

1.3 Events