Contents

1 Introduction

Writing algorithms which can answer questions on pictures or videos with a high level of accuracy and generality has been a goal of researchers in the AI community for many years. Recently, a lot of progress has been made in this area; with advances in neural network models and the production of larger datasets allowing reseachers to significantly improve accuracy on question answering models.

Formally, Visual Question Answering (VQA)[3] is a task where, given an image and a question posed in natural language about the image, a model is required to produce an open-ended answer to the question. Video Question Answering (VideoQA) is a related task where a model is given a video (mulitple images in sequence) and a question. These questions can be related to a single frame of the video, effectiively making VideoQA a superset of the VQA task.

This project will attempt to produce a hybrid model for VideoQA - one which makes use of both neural networks and knowledge representation and reasoning methods based on first-order logic.

1.1 Motivation

VQA and VideoQA tasks attract attention because of their difficulty; both problems are considered "AI-complete" - they require knowledge from multiple modalities beyond a single sub-domain[2]. Building systems which have a deep understanding of the world would be a significant achievement for AI research; allowing many tasks which require significant human time and effort to be automated. Solving the VideoQA problem, which require image understanding, natural language understanding and commensense reasoning to be deployed, could be a major step towards this.

Furthermore, subproblems of VideoQA have already been shown to have applications in real-world tasks, for example event recognition has been used for identifier attacks on computer networks[5], detecting credit card fraud[16] and recognising cardiac arrhythmias[12].

Finally, the use of a hydrid model for VideoQA brings with it a nmuber of advantages. Firstly, representing knowledge in logical form allows the injection of commensense or background knowledge which can significantly improve accuracy in question answering tasks[11]. Secondly, there has recently been a significant increase in research related to explainable AI methods - machine learning techniques that enable human users to understand, trust and manage emerging artificially intelligent partners[4]. Extracting the knowledge from a neural network into logical form could be an important step toward explaining and understanding their behaviour.

1.2 Problem Description

As mentioned above, the VideoQA problem can be defined as building a model which, when presented with a short video and an open-ended, natural language question about the video, can produce a natural language answer to the given question. In our case we are looking to design a hyrid model for VideoQA. More specifically, convolutional neural networks (CNNs) will be used to extract knowledge from each frame of the video. This knowledge, along with the question to be answered, will be represented in a fashion that is amenable to searching for the answer to the question. This framing of the problem leads us to outline the following sub-problems, which will need to solved in order to produce a satisfactory VideoQA model.

1. **Object Detection**. Given a frame, we need a model which can produce a rough estimate (a bounding box, for example) of the location of an object in the frame. We will also need a model which can classify each detected object into a set of predefined classes.

2. **Property Extraction**. Given an object, which is the output of the 'object detection' model above, we need an algorithm which can produce a set of values for that object for some set of predefined properties. For example, we might need to give a value for the colour, size or shape of an object.

3. **Event Detection**. Given two sequential frames, we need a model which can classify the event(s) which occurred between the two frames into a set of predefined classes (possibly including a catch-all 'no event' class). This model will also be required to list objects involved in the event and what their role in the event was. This will require some level of object tracking so that it is clear how objects are related between frames.

4. **Question and Knowledge Representation**. Given the outputs of the models above and the natural language question, we need a way of representing the background knowledge, the question and the knowledge contained in the frames of the video. These must be represented in a manner that allows an answer to the question to be found.

1.3 Objectives

A lot of research has been conducted on building end-to-end neural network models to solve VideoQA tasks (see section 3 for examples), but very little prior work has been done on hybrid models. The main aim of this project, therefore, is to explore the possibility of adding logical representation of knowledge to existing deep learning techniques. The following are the primary objectives of the project:

1. Construct a hyrid VideoQA model which allows injection of background knowledge and helps to increase the explainability of the model.
2. Find a challenging dataset for training the model (or construct a dataset if none are suitable)
3. Compare qualitatively (and quantitatively, if possible) existing approaches to our own hyrid model.
4. Investigate the possibility of learning domain-dependent logical rules, rather than having to have them hard coded for each environment.

2 Background

This section introduces some technical background which will likely be required as part of the project. It includes an introduction to neural networks and CNNs, a comparison of some existing object tracking and object detection algorithms and a discussion on knowledge representation and reasoning and symbolic rule learning.

2.1 Deep Learning

Deep neural networks (DNNs) have emerged as a very successful algorithm for machine learning; deep learning has been used to beat records in tasks such as image recognition, speech recognition and language translation[9]. Many different architectures have been proposed to solve various tasks, these architectures include convolutional neural networks (CNNs), which are designed to process data that come in the form of multiple arrays[9], and recurrent neural networks (RNNs), which are designed to process sequences of arbitrary length[10]. The following section gives a brief introduction to CNNs and describes some of their use cases.

2.1.1 Convolutional Neural Networks

CNNs contain three types of layers: convolution, pooling and fully connected. Units (artificial neurons) in a convolution layer are organised into feature maps. The inputs to each unit in a feature map come from the outputs of the units in a small region of the previous layer, the output of the unit is then calculated by passing the weighted sum of its inputs through an activation function such as ReLU. The set of weights, also known as a filter or kernel, is the part of the layer which is learned through backpropagation. Every unit in a feature map has the same kernel. Each feature map in a layer has its own kernel. Pooling layers reduce the size of the input by merging multiple units into one. A typical pooling operation is max-pooling, which computes the maximum of a local patch of units. Finally, in fully-connected

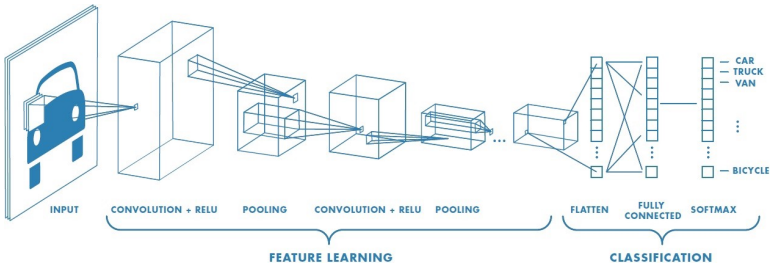

Figure 1: An example of a CNN architecture. The input image is passed through a series of convolution and pooling layers before being flattened into a one-dimensional layer and passed through one final fully connected layer. The softmax classification function is then applied at the output.

layers (which are typically placed at the output of the CNN) every unit in a layer is connected to every unit in the previous layer. An example CNN architecture is shown in figure 1.

CNNs have proven to be adept at a number of tasks involving images, including image classification[8] and object detection[13][15]. We explore these further in Section 2.2.

2.2 Image Processing

2.2.1 Object Detection

The object detection task could formally be defined as designing a model which, when given an image, can produce a rough localisation of objects of interest in the image (in the form of a bounding box) and classify eac of these objects into a set of predefined classes. In this section we introduce two well known object detection algorithms, Faster R-CNN[15] and You Only Look Once (YOLO)[13].

Faster R-CNN is an evolution of previous object detection algorithms, R-CNN[7] and Fast R-CNN[6]. Faster R-CNN builds on its predecessors by adding a region proposal network (RPN) - a neural network which takes an image and produces a set of region of interest (RoI) proposals. This method of region proposal is much faster than previous algorithms (such as those used in [7] and [6]) since it is able to make use of the GPU, as opposed to requiring the CPU. Faster R-CNN then uses a similar classifier and bounding box regressor as Fast R-CNN at the output; this section of the network also receives the feature maps from the final layer of the RPN, in this sense the initial layers of the network are shared between the region proposal section and the classifier/regressor section. A diagram of the Faster R-CNN architecture is shown in figure 2.

The three object detection algorithms mentioned above all work by first producing region proposals, then producing a more accurate localisation and

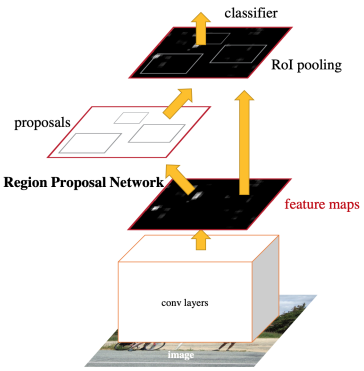

Figure 2: Diagram of the Faster R-CNN architecture. One part of the network produces region proposals and another part handles bounding box regression and object classification. Figure from [15].

a score for each region and finally removing any low-scoring or redundant regions. This requires the algorithm to 'look' at the image multiple times (around 2000 times for R-CNN). You Only Look Once (YOLO) is a significantly more time-efficient algorithm which, as the name suggests, takes a single look at the image. A convolutional neural network is used to simultaneously predict multiple bounding boxes and the class probabilities for each box. As well as being very fast, YOLO makes fewer than half the number of background errors (where the algorithm mistakes background patches for objects) as Fast R-CNN [14]. YOLO is, however, slightly less accurate than some of the slower methods for object detection[13].

2.2.2 Commonly Used Metrics

In this section we present some commonly used metrics for classification and object detection tasks. We use TP, TN, FP and FN to mean True Positive, True Negative, False Positive and False Negative, respectively.

Firstly, for classification tasks the following terminology is commonly used:

- $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$. The accuracy is the ratio of correct predictions to the total number of predictions.
- $Precision = \frac{TP}{TP+FP}$. The precision is the ability of a classifier to not label the negative data as positive.
- $Recall = \frac{TP}{TP+FN}$. The recall is the ability of the classifier to find the positively-labelled data.
- $F_1 = 2 * \frac{precision*recall}{precision+recall}$. The F_1 score is a way of combining the precision and recall scores.

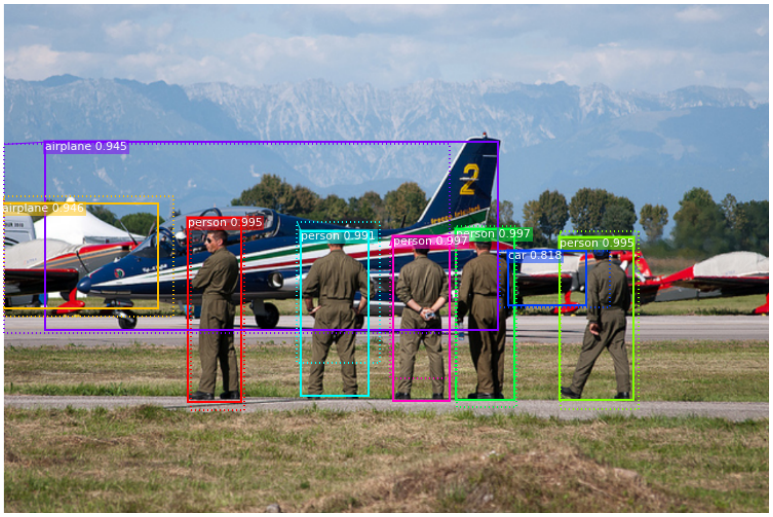

Figure 3: An example of the bounding boxes and confidence scores produced by an object detection algorithm. Image from [1].

Each object detector model will output a confidence score for each object classification it makes. We can then set a threshold value which determines what is counted as a classification of an object. Altering this threshold value will give different precision and recall values for the model, which can then be plotted on a precision-recall graph. Example precision-recall curves are shown in figure 4.

For object detection tasks, where a bounding box is produced as a rough localisation of an object's position, metrics which measure the accuracy of the localisation of an object are required. One very common metric is the Average Precision (AP), which is roughly defined as the area under the precision-recall curve (estimates of this value are usually used for competition datasets).

2.2.3 Object Tracking

2.3 Knowledge Representation and Reasoning

2.3.1 The Event Calculus

2.3.2 Answer Set Programming

2.3.3 Symbolic Rule Learning

3 Related Work

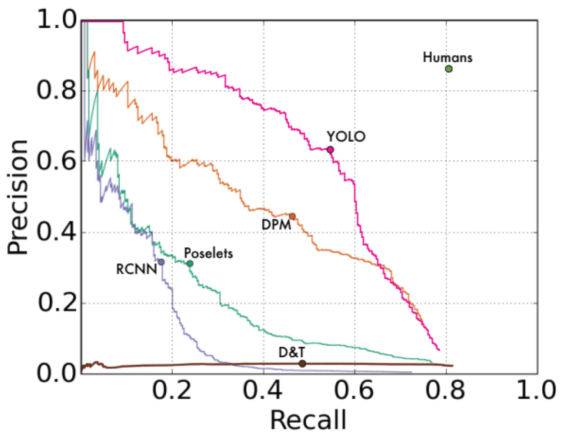

Figure 4: Example precision-recall curves for various object detection models. Image from [14].

References

[1] W. Abdulla. *Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow*. URL: `https://github.com/matterport/Mask_RCNN` (visited on 01/15/2020).

[2] Somak Aditya, Yezhou Yang, and Chitta Baral. "Explicit Reasoning over End-to-End Neural Architectures for Visual Question Answering". In: *CoRR* abs/1803.08896 (2018). arXiv: `1803.08896`. URL: `http://arxiv.org/abs/1803.08896`.

[3] Aishwarya Agrawal et al. "Vqa: Visual question answering". In: *International Journal of Computer Vision* 123.1 (2017), pp. 4–31.

[4] Alejandro Barredo Arrieta et al. *Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI*. 2019. arXiv: `1910.10045 [cs.AI]`.

[5] Christophe Dousson, Pierre Le Maigat, and France Telecom R&d. "Chronicle Recognition Improvement Using Temporal Focusing and Hierarchization". In: *IJCAI*. 2007, pp. 324–329.

[6] Ross Girshick. "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[7] Ross Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., 2012, pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[9] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521 (2015), pp. 436–444.

[10] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. "Recurrent neural network for text classification with multi-task learning". In: *arXiv preprint arXiv:1605.05101* (2016).

[11] Kenneth Marino et al. "OK-VQA: A Visual Question Answering Benchmark Requiring External Knowledge". In: *CoRR* abs/1906.00067 (2019). arXiv: 1906.00067. URL: http://arxiv.org/abs/1906.00067.

[12] Ren Quiniou et al. "Intelligent Adaptive Monitoring for Cardiac Surveillance". In: *Computational Intelligence in Healthcare 4: Advanced Methodologies*. Ed. by Isabelle Bichindaritz et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 329–346. ISBN: 978-3-642-14464-6. DOI: 10.1007/978-3-642-14464-6_15. URL: https://doi.org/10.1007/978-3-642-14464-6_15.

[13] Joseph Redmon and Ali Farhadi. "Yolov3: An incremental improvement". In: *arXiv preprint arXiv:1804.02767* (2018).

[14] Joseph Redmon et al. "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[15] Shaoqing Ren et al. "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[16] Nicholas Poul Schultz-Møller, Matteo Migliavacca, and Peter Pietzuch. "Distributed Complex Event Processing with Query Rewriting". In: *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*. DEBS 09. New York, NY, USA: Association for Computing Machinery, 2009. ISBN: 9781605586656. DOI: 10.1145/1619258.1619264. URL: https://doi.org/10.1145/1619258.1619264.