R-FCN Object Detection Ensemble based on Object Resolution and Image Quality

Christoffer Bøgelund Rasmussen[1] and Kamal Nasrollahi[1]
[1]*Visual Analysis of People (VAP) Laboratory, Aalborg University, Aalborg, Denmark*
cbra12@student.aau.dk, kn@create.aau.dk

Keywords: Convolutional Neural Networks, Object Detection, Image Quality Assessment, Ensemble Learning.

Abstract: Object detection can be difficult due to challenges such as variations in objects both inter- and intra-class. Additionally, variations can also be present between images. Based on this, research was conducted into creating an ensemble of Region-based Fully Convolutional Networks (R-FCN) object detectors. Ensemble methods explored were firstly data sampling and selection and secondly combination strategies. Data sampling and selection aimed to create different subsets of data with lowered variance with respect to object size and image quality such that expert R-FCN ensemble members could be trained. Two combination strategies were explored for combining the individual member detections into an ensemble result. R-FCNs were trained and tested on the PASCAL VOC benchmark object detection dataset. Results proved positive with an increase in AP when ensemble members were combined appropriately. The method shows potential and other object or image variations could be sampled to see if a more robust ensemble could be made.

1 INTRODUCTION

Object detection is a fundamental area of computer vision that has had a great amount of research over the past decades. The general goal of object detection is to find a specific object in an image. The specific object is typically from a pre-defined list of categories that are of interest for a given use case. Object detection generally consists of two larger tasks; localisation and classification. It is assumed that the objects of interest are not already located in the image and as objects can vary in number of pixels depending on factors such as distance and scale. Localisation is typically done with a bounding-box indicating where a given object is in the image. However, other methods such as objects' centres and closed boundaries can also be used (Zhang et al., 2013). Not only is object detection an important task in localising and classifying, it is also a necessary earlier step in larger computer vision pipelines. For example, object detection is needed within the tasks such as activity and event recognition, scene understanding, and robotic picking.

Object detection is a challenging problem due to both large scale issues and minute differences between objects. Firstly, there is the challenge of differentiating objects between classes. Depending on the problem at hand the number of potential classes present can be into the thousands or tens of thousand. On top of this, separate object categories can be both very different in appearance, for example an apple and an aeroplane, but separate categories can also be similar in appearance, such as dogs and wolves. These main challenges of object detection stem from two categories which as defined per (Zhang et al., 2013) as: robustness-related and computational-complexitity and scalability-related.

Robustness-related refers to the challenges in appearance variations within the both of intra-class and inter-class. Intra-class is the differences in appearance of objects which are of the same class. An object detection system must be able to learn the appearance variations that can occur intra-class. These can be categorised into two types as per (Schroff, 2009) as object and image variations. Object variations consist of appearance differences between object instances with respect to factors such as colour, texture, shape, and size. Image variations are differences not related to the object instances themselves but rather the actual image. This can consist of conditions such as lighting, viewpoint, scale, occlusion, and clutter. Based upon these differences the task of both classifying a given object as a given class but also differentiating the potentially largely varying objects into the same class is challenging.

Current state-of-the-art in object detection is

within the realm of deep learning with CNNs. Deep learning methods are of such a scale that given appropriate data have been able to address the two main challenges mentioned earlier. This is exemplified with almost all leading entries in benchmark challenges such as PASCAL VOC (Everingham et al., 2010), ImageNet (Russakovsky et al., 2015), and MSCOCO (Lin et al., 2014) consisting of CNN-based approaches. Additionally, recent trends with Convolutional Neural Network (CNN)-based object detection methods have been to incorporate ensembles of networks to further enhance performance.

One of the main goals of an ensemble system is to reduce the variance incorporated in the training process. An example is to train classifiers on different subsets of the data, creating a number of different ensemble members. The assumption is that the classifiers will make different errors on a given data point. However, by combining the classifiers the errors will be cancelled out by the increased strength from lower individual variance. The ensemble members created in this work followed the three main strategies from (Zhang and Ma, 2012) to build an ensemble system. Namely:

1. Data sampling and selection: selection of training data for individual classifiers.
2. Training member classifiers: specific procedure used for generating ensemble members.
3. Combining ensemble members: combination rule for obtaining ensemble decision.

The firstly strategy aims to increase the diversity of the individual ensemble members. A common method as mentioned earlier, is to train the members on different subsets of the training data creating members that have different strengths and weaknesses in the overall data. The second strategy is in regards to how the members are trained. Variability in the ensemble can be reduced by using different strategies. This could be by altering the inner parameters of a CNN such as loss functions, filter sizes and optimisation strategies Third is the final step in an ensemble system, how to combine the members. For example, if an object detection system outputs bounding boxes with accompanying confidence values these can be combined multiple arithmetic methods. Such as mean, average, minimum, maximum and median.

This work addresses the robustness-related challenges by exploring the possibilities of designing expert ensemble members towards both object and image variations in a leading object detection benchmark. This is done by training an ensemble of Region-based Fully Convolutional Network (R-FCN) with ResNet-101 networks. Data sampling strategies are used to create subsets of data with respect to object resolution and various image quality factors. Finally, two separate combination strategies are explored for combining the ensemble members.

2 RELATED WORK

One of the first methods to show that CNNs could significantly improve object detection was that of R-CNN (Girshick et al., 2014). The method obtains the name R-CNN based upon a CNN is used on regions of the image. Many earlier object detection approaches were used in a sliding window fashion testing all areas of an image. This can lead to a huge amount of potential testing windows especially if the object detection is done at a multitude of different scales. In R-CNN the CNN model is used as a feature extractor from which a class-specific linear Support Vector Machine (SVM) can be trained on top of. The AlexNet-based feature extractor is firstly pre-trained on a large dataset designed for classification, in this case the training set from ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012. This pre-trained model is then adapted to the new domain of object detection by fine-tuning the model accordingly. The pipeline of the R-CNN is split into 3 modules, region proposals, feature extraction, and class-specific linear SVMs. The first module aims to reduce the amount of classification windows in comparison to a sliding window approach. R-CNN is agnostic to the region proposal method chosen, and in the original work SelectiveSearch (Uijlings et al., 2013) is used. Module two, is the use of a CNN as a feature extractor. This is in the form of a 4096-dimensional feature vector from the domain-specific Pattern Analysis, Statistical Modelling and Computational Learning Visual Object Classes (PASCAL VOC) trained AlexNet model. These feature vectors are used in the third module, a class-specific linear SVM is trained to classify proposals.

The R-CNN method was improved the following year with Fast R-CNN (Girshick, 2015) and aimed to improve speed and accuracy. One of the significant changes is that the detection training done is now end-end rather than in the multi-stage pipeline in R-CNN. The Fast R-CNN method takes both an image and a set of pre-computed object proposals, as in R-CNN. A CNN forward propagates the entire image, rather than individual proposals in R-CNN, through several convolutional and max-pooling layers to produce a feature map. Features are extracted for each proposal in their corresponding location in the computed feature map with a Region of Interest (RoI) pooling layer.

Once the RoI pooling layer has been applied to a proposal the forward pass continues through two fully-connected layers followed by two sibling output layers. The sibling outputs are a softmax classification layer that produces probabilities for the object classes and another layer for bounding-box regression. These two layers replace the respective external modules in R-CNN and make it possible to train the entire detection network in a single-stage. In R-CNN, the only deep network used was AlexNet (Krizhevsky et al., 2012), however, in Fast R-CNN the authors experiment with networks of different size. It was found that the deeper network VGG-16 (Simonyan and Zisserman, 2015) for computing the convolutional feature map gave a considerable improvement in performance.

However, as the name Fast R-CNN implies the main improvement is the speed in respect to both training and testing. By computing a convolutional feature map for an entire image rather than per object proposal the number of passes in the network is lowered significantly. While Fast R-CNN provided improvements in both accuracy and speed, the increase in speed is only in relation to the actual object detection and assumes that the region proposals are precomputed. Therefore, there is still a significant bottleneck per image as a region proposal method can typically take a couple of seconds.

Faster R-CNN (Ren et al., 2015) addressed the region proposal bottleneck in the third iteration of the R-CNN method. Faster R-CNN showed that region proposals could be computed as part of the network through the use of a Region Proposal Network (RPN). The RPN shares the convolutional layers and feature map used for computing features with RoI pooling in Fast R-CNN. As these layers are already computed on the entire image for the classification pipeline, the added time for proposals using the RPN is negligible in comparison to a method such as SelectiveSearch. Apart from the change in how region proposals are computed, there is no difference in comparison to Fast R-CNN. An RPN takes the last convolutional feature map as input and returns a number of object proposals. Each proposal is fed into two sibling layers, similar to that in Fast R-CNN, one layer scoring how likely to be an object or background and another performing bounding-box regression. Once the proposals have been found with the RPN they are placed on the same convolutional feature map as earlier and the rest of the pipeline is identical to Fast R-CNN, classifying and regressing bounding-boxes with another set of sibling layers. As the only change is the addition of computing proposals in the network with RPN, the results are similar in respect to Average Precision (AP). The main contribution to the work is the speed-up of the entire object detection pipeline as the object proposal time is now minimal. A speed-up of $10\times$ from Fast R-CNN to Faster R-CNN and $250\times$ from the original R-CNN.

Much of the recent work within object detection has been based upon the Faster R-CNN framework. This is exemplified by looking at the Microsoft Common Objects in Context (MS COCO) detection leaderboard (COCO, 2017), with 15 of the 21 approaches being Faster R-CNN related as of early 2017. Firstly, the winner of the MS COCO 2015 and ILSVRC 2015 detection challenge was with the use of deep residual networks (ResNets) (He et al., 2015). As is well known with CNNs, deeper networks are able to capture richer higher-level features. The authors showed that this is also beneficial in the object detection domain. In (He et al., 2015) an ensemble of three deep residual networks with 101 layers was trained for object detection and another ensemble of three used for region proposals with the RPN while being based on the Faster R-CNN framework. In addition to the ensemble, the winning entry also added box refinement, global context, and multi-scale testing to the Faster R-CNN.

The current leading method on MS COCO is an extension of the previously explained ResNets (He et al., 2015). This method dubbed G-RMI on the MS COCO leaderboard (COCO, 2017) is an ensemble of five deep residual networks based upon ResNet (He et al., 2015) and Inception ResNet (Szegedy et al., 2016) feature extractors. No work has been published yet on G-RMI at this time, however, a short explanation of the entry is included in a survey paper from the winning authors (Huang et al., 2016). The approach was to train a large number of Faster R-CNN models with varying output stride, variations on the loss function, and different ordering of the training data. Based upon the collection of models, five were greedily chosen based upon performance on a validation set. While performance on the models were important, the models were also chosen such that they were not too similar. It should also be noted that apart from the ensemble of models, G-RMI did not include any extras such as multi-scale training, box refinement, or global context which are often used in benchmark challenge entries.

Recently, a newer approach to region-based methods has been proposed with the use of Fully Convolutional Networks (FCNs) through the R-FCN (Dai et al., 2016). The overall approach is similar to that used in region-based methods such as (Girshick et al., 2014), (Girshick, 2015) and (Ren et al., 2015). First compute RoIs using a region proposal method and

second perform classification on these regions. R-FCN uses the RPN from Faster R-CNN (Ren et al., 2015) for class-agnostic RoI computation. However, rather than extracting features with RoI-pooling, fully convolutional position-sensitive score maps are computed. The score maps are split up to represent a relative position in a $k \times k$ grid, with each cell presenting information relative to the spatial position of an object. For example, the upper-left cell represents scores that pixels are present at that relative position to the object. A bank for position-sensitive score maps are found for each class. After computing the bank, the R-FCN computes a position-sensitive RoI-pooling layer for each class. For each RoI found with the RPN each cell aggregates the response from the appropriate score map from the bank of maps. While the ordering and methodology of RoI-pooling is different in R-FCN to that of Faster R-CNN the same backbone CNN can be used. In the experiments conducted by the authors a ResNet-101 network is chosen. Overall R-FCN is an improvement on the Faster R-CNN approach on benchmarks such as PASCAL VOC and MS COCO. It is also competitive with the MS COCO 2015 winning entry (He et al., 2015), while not having any additions such as global context or iterative box regression. Additionally it is considerably faster in training and testing in comparison to Faster R-CNN.

The use of FCNs is currently the leading method for segmentation; both semantic (Long et al., 2014) and instance (Li et al., 2016). The latter, named Fully Convolutional Instance-aware Segmentation (FCIS) won the 2016 MS COCO instance segmentation challenge and is also the current second place in bounding box object detection. It uses a similar approach with position-sensitive score maps for pixel-level likelihood for an object category to produce bounding boxes. From these, instance segmentation is performed to produce the pixel-level classification. The main differences between R-FCN and FCIS is the addition of ensemble of ResNet models, multi-scale testing and training, and horizontal flipping.

1

2

[1] FiXme Note: Include more related about ensembles

[2] FiXme Note: add overview of key results on benchmarks

3 OBJECT DETECTION WITH R-FCN

One of the current leading object detection methods is the R-FCN (Dai et al., 2016), which as mentioned in Section **?? ??**, takes a different approach to that of the region-based methods such as Faster R-CNN. The authors of R-FCN were inspired by the recent advances in FCN classification networks, such as ResNets. They argue that the addition of the RoI-pooling layer in the Faster R-CNN pipeline is unnatural and adds computational complexity. The hypothesis is that the reasoning behind this addition is due to the trade-off between using a classification approach in an object detection pipeline. A defining factor in object detection is that the method should be able to respect translation variance, that translation of an object inside an object proposal should given a good indication as to how well the proposal fits the object. Whereas classification is more translation invariant, as the shifting of an object in an image does not effect how the system returns its output. The use of the RoI-pooling layer placed in between convolutional layers means that any convolutions after this point are not translation invariant as it is not region specific. Rather than using this popular feature extractor, R-FCN uses position-sensitive score maps computed by a bank of convolutional layers. The maps add translation variance into the detection pipeline by computing scores in relation to position information with respect to the relative spatial position of an object. A RoI-pooling layer is added after the score-maps, however, no convolutional operations are done after this point ensuring translation variance.

The overall approach of the R-FCN also consists of the popular two-stages of region proposal and region classification. Region proposal is done using the RPN from Faster R-CNN followed by the position-sensitive score maps and RoI pooling for region classification. The overall architecture of the R-FCN can be seen in Figure 1. Similar to Faster R-CNN, convolutional layers are applied on the input image and the RPN computes region proposals. After this position-sensitive score maps aid in classification.

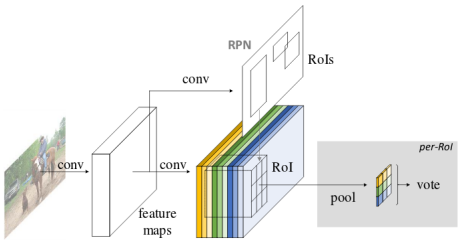


Figure 1: Architecture of R-FCN. Region proposals are found using the RPN followed by classification based on a bank of position-sensitive score maps (Dai et al., 2016).

The added translation variance post finding proposals with the RPN is done by producing a bank of k^2 score maps for each object category. Therefore, there are a total of $k^2(C+1)$ maps. The number of k^2 maps is due to a $k \times k$ spatial grid representing relative positions. Typically $k = 3$, therefore, nine score maps represent position-sensitive scores for a given object category. This is illustrated in Figure 2, each of the 9 coloured rectangles on the left of the figure represent the k^2 score maps. Each colour represents one of the relative positions. For example, the three shades of blue are positions in the bottom of a RoI, where the darkest is bottom-right, then bottom-centre and lightest bottom-right. For a given RoI placement the vote for relative position is sampled from their respective map in the bank.

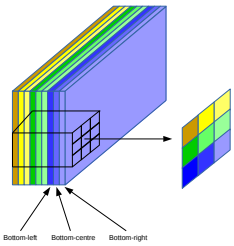


Figure 2: A bank of score maps are present for each object category. For a given RoI, the score is sampled from the respective position in the corresponding score map.

Once the bank of score maps have been computed, position-sensitive RoI-pooling is found for region classification. Each individual $k \times k$ bin pools from its corresponding location in the relevant score map. For example, the top left bin pools from that position in the top-left score map and so on. The RoI-pool is computed using average pooling for each bin which can be seen in Figure 3. The final decision for a given class is determined by a vote where each of the bins are averaged, producing a $(C+1)$-dimensional vector for each RoI.

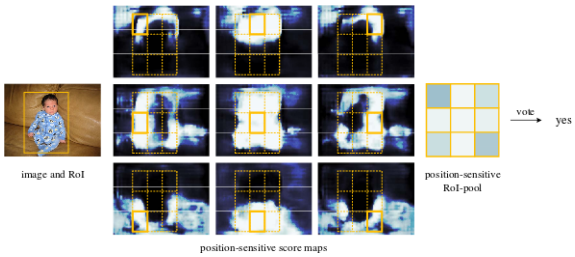


Figure 3: Position-sensitive RoI-pooling operation for a given class (Dai et al., 2016).

4 PROPOSED METHOD

Now that an analysis of the technical aspects of object detection with deep learning has been conducted an overview of the design of the system will be made in this section. Multiple choices can be made with respect to the overall architecture of the CNN-based object detector. As covered in Section **?? ??**, two of the best performing systems are Faster R-CNN and R-FCN. The current core classification model used in both is the ResNet architecture. As the addition of ResNets significantly increases performance the use of these in this work is deemed crucial. However, the choice of either Faster R-CNN or R-FCN is not immediately as clear. Both methods perform similarly with respect to benchmarks such as PASCAL VOC and MS COCO. But as the decision has been made to incorporate ResNets a decision on this matter was indirectly made. The GPU available in this project while being large in regards to memory is only available to train R-FCN with the ResNet-101 model. Unfortunately, due to the internal architecture of Faster R-CNN the 8GB memory on the NVIDIA GPU was not able to store all parameters while training a Faster R-CNN with ResNets. Due to the more efficient classification module in R-FCN, a ResNet backbone could be trained.

Leading object detection systems take advantage of ensemble methods. Many of them are trained with regards to the variations in internal architecture and not specifically training experts towards solving specific challenges. Therefore, the system in this project will take advantage of the first point in Section **?? ??**, namely data sampling and selection. The aim will be to train R-FCN with ResNet-101 on different subsets of training data with the aim to create expert ensemble members in regards robust-related challenges. Two separate factors will be chosen, one with respect to variations in the object and the other in terms of image variations. The first factor chosen is object size. As covered in Section **?? ??**, in general object detection systems find it challenging to detect and classify

objects with smaller resolutions. Therefore, if a system can be trained towards a subset of sizes in the training data, ideally the individual ensemble members will increase their performance on the respective sizes. The second factor chosen is image quality. As mentioned in Section **?? ??**, the quality of an image can be a factor in the overall performance of CNN-based classification systems. Therefore members will also be trained towards subsets data split based upon this.

Lastly, the third strategy in building an ensemble system will be addressed. Individual members predictions much be combined in an ensemble system. Therefore, approaches must be taken to combine outputs. The combination strategy is greater than only voting on which class a potential object is associated to. Bounding-boxes and the confidence of each detection is used in the calculation of metrics in both PASCAL VOC and MS COCO. Therefore, these must be combined in the ensemble system as well.

Based upon these issues the following design requirements are set with respect to the previously discussed items.

- Object Detector Architecture.
  - CNN-based method.
  - ResNets as backbone model.
- Ensemble Data Sampling and Selection.
  - Ability to measure object and image variations with respect to:
    - Object size.
    - Image quality.
- Ensemble Training of Classifiers.
  - Must be kept constant to measure effect of differing data sampling strategy.
- Ensemble Combination.
  - Method to combine bounding-boxes and confidences between individual ensemble members.

Now that the general requirements have been outlined the following sections will cover the architectural considerations to ensure that the above requirements can be met.

4.1 Training Ensemble Members

The training of the R-FCN members will be done using Convolutional Architecture for Fast Feature Embedding (Caffe) (Jia et al., 2014). This was chosen due to the research being provided by the authors of R-FCN through training code and pre-trained Caffe models. However, as there is the requirement to combine detections between ensemble members, then the detections must be found based upon the same input to each model. One solution to this is to use the region proposals found using the RPN. In a standard R-FCN the RPN is an internal part of the network and is trained end-to-end. However, as these proposals must be constant between all ensemble members this method is not appropriate. Additionally, due to the nature of the Caffe framework, once a network has been defined and trained it is difficult to change it. For example, a standard R-FCN takes the entire image as inputs but in this work the requirement is that it takes smaller region proposals. The solution to these points is to train the networks using a method inspired by the 4-step alternating training method presented by the Faster R-CNN authors (Ren et al., 2015). The process can be seen in Figure 4.

In this approach the overall network in trained in multiple steps rather than an end-to-end method. In the first step, an RPN is trained to determine region proposals, the RPN is initialised from a pre-trained ImageNet model and fine-tuned to the proposal task. Next a R-FCN is trained based upon the proposals found in the previous step. This network is also initialised with a pre-trained ImageNet model. In step three, another RPN is trained but initialised using the R-FCN from step two. In this step the convolutional layers that are shared between the R-FCN and RPN are fixed and only the layers unique to the RPN are updated. By training a model with this approach a testing image is able to run through the same steps as a R-FCN trained end-to-end, however, as the networks are split into different models it is also possible to use the stages of the method individually. Creating a solution for finding region proposals with an external RPN and having a R-FCN that can take the proposals as inputs.

An additional benefit to training R-FCNs in this manner is that once a baseline model has been created only one part needs to be re-trained. As the aim is to train various ensemble members to different subsets of data only the R-FCN in stage 2 is required to be re-purposed. The RPN in stage 3 should be kept constant based on the baseline model as it will provide the shared proposals for test images. Therefore, once a systematic approach has been found for splitting data for both train and test based on the data sampling and selection requirements the detection part of the R-FCN can be trained towards its expert area. The following sections will explain how the subsets of data will be selected.

4.1.1 Object Size Data Sampling

The area of a region proposal found with a RPN gives an indication as to the approximate size of a poten-

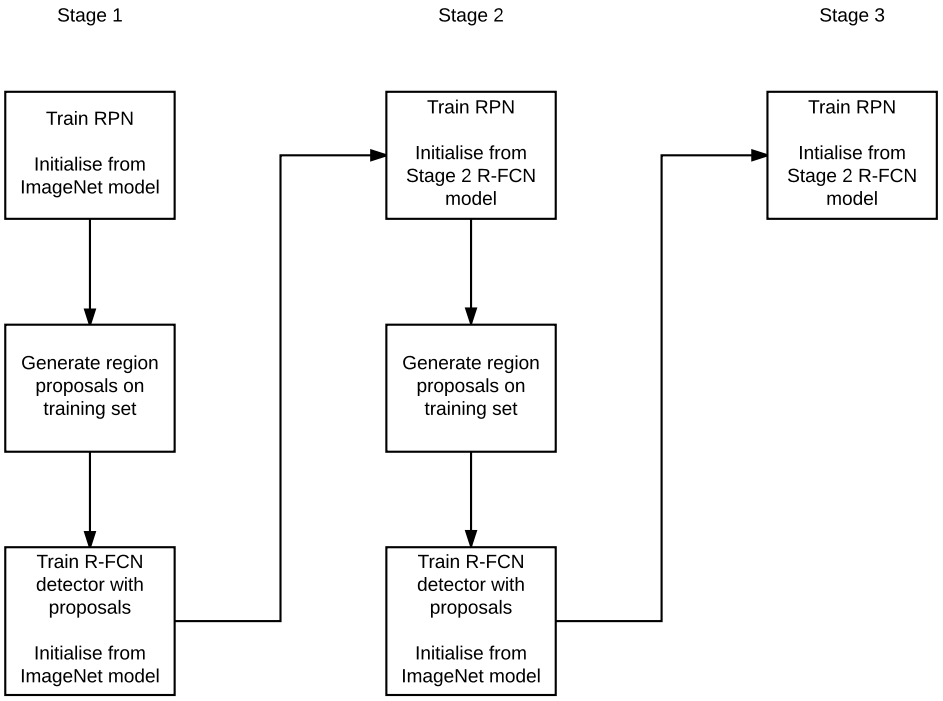


Figure 4: Flow chart showing the alternating training method.

tial objects. Therefore, the area for all proposals on the training set can be computed from the output of the second step in stage 2 shown in Figure 4. Once the area of all proposals are computed an appropriate split of the data can be determined depending on the area distribution. The main requirement in creating the subsets of data is that equal number of ground truth samples should be present in both.

4.1.2 Image Quality Data Sampling

There are many choices for computing the quality of an image. A popular area of research for this purpose is Image Quality Assessment (IQA). These methods aim to determine the subjective quality of an image. There are two forms of IQA, Full-Reference Image Quality Assessment (FR-IQA) and No-Reference Image Quality Assessment (NR-IQA). FR-IQA approaches require the original, undistorted reference image in order to determine quality. Whereas, NR-IQA do not have this information available (Bosse et al., 2016). As the aim is to determine the level of image quality on one of the benchmark datasets, no reference image is present. Therefore, an NR-IQA method is required. Current state-of-the-art within NR-IQA is also deep learning based and works are typically trained on IQA datasets. Datasets include Laboratory for Image & Video Engineering (LIVE) dataset (Sheikh et al., 2006) (Sheikh et al.,), TID2013 (Ponomarenko et al., 2013) and CSIQ (Larson and Chandler, 2009). The datasets consist of source reference image and have artificially created counterparts with varying levels of distortion. Distortions include, such as in the LIVE dataset, JPEG2000 compression, JPEG compression, additive white Gaussian noise, Gaussian blur and bit errors from a fast fading Rayleigh channel. Models can then be trained to predict subjective quality based on ground truth user determined quality measurement.

Based upon this, an NR-IQA method can be used to determine the level of image quality with respect to a number of different distortions. Then as in object size training, if possible, the data will be split into different training subsets. A leading CNN-based NR-IQA method will be used and the specific network and implementation details will be covered in Section **?? ??**.

4.1.3 R-FCN Training

Training of the baseline R-FCN model shown in Figure 4 is done using Stochastic Gradient Descent (SGD) optimisation with largely the same parameters across the five different training parts. The parameters are adapted from (Dai et al., 2016) and can be seen in Table 1. All models start with a base learning rate of 0.001 which is dropped by a factor of 0.1 once in the process. This is done after 80,000 iterations for the

R-FCN models and after 60,000 for the RPNs. The learning rate is controlled with a momentum of 0.9 and weight decay of 0.0005. The two R-FCN models are trained for 120,000 iterations, while the three RPNs are trained for 80,000.

Table 1: Common SGD optimisation parameters for the 5 training parts of the baseline R-FCN model.

Parameter	**Value**
Base learning rate	0.001
Learning rate policy	step
Gamma	0.1
Momentum	0.9
Weight decay	0.0005

Both networks are trained with a batchsize of one example per iteration. In the RPN models the batches are simply one ground truth example per iteration. Whereas, the training of the R-FCNs sample 128 mini-batches from a given image. These mini-batches can consist of both object class samples and background samples. The only data augmentation used in training is horizontal flipping of images, effectively creating double the amount of training examples.

4.2 Object Detection Benchmark

The choice of object detection benchmark is PASCAL VOC. A common strategy in object detection systems is the conduct preliminary experiments on PASCAL VOC data. Then if applicable a potential system can be trained on a larger dataset such as MS COCO. By training on PASCAL VOC a larger number of ensemble members can be trained on the single GPU available.

The data used will follow the leading methods for PASCAL VOC 2007 object detection. Training will be done on the 07+12 train sets and testing will be conducted on the 07 test set.

4.3 Resolution-Aware Detection

As determined earlier object detectors are generally more accurate on objects that cover a larger number of pixels in an image. This is intuitive as objects with a lower resolution objectively have less details that can describe them. The poorer performance can be seen in Table **??**, for all object detectors the AP is considerably lower for smaller objects in comparison to both medium and large. The best performing detector from (He et al., 2015), has an AP difference of 35.3%, from 50.9% for large objects to 15.6% for small. A potential method of tackling this issue is to train multiple detectors on separate partitions of the training data according to the size of the object. While deep-based CNN have millions of parameters to generalise from training to tesitng, the difference between small and large objects may skew the learning towards the latter. In order to test this hypothesis an initial test will be conducted on the 07+12 train set. However, PASCAL VOC does not have the same definition of objects sizes as in MS COCO. Therefore, the distribution of the ground truth bounding boxes from the 07+12 set must be analysed in order to determine if an appropriate split of data based on object size can be made. This was done by parsing all of the bounding box coordinates in the set and calculating the area. A histogram of the all of the ground truth areas can be seen in Figure 5. There is a clear tendency to smaller objects in the training set with a clear skew towards the left of the figure.

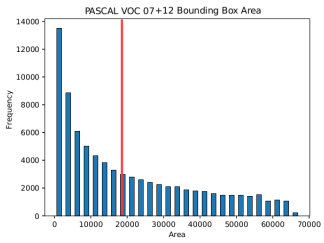


Figure 5: Histogram of the PASCAL VOC 07+12 bounding box area.

The data in Figure 5 can be split into two equal subsets if the median area of 19,205.5 is used. However, the ensemble R-FCN members are trained with region proposal inputs of both ground truth positives and negative examples found using a RPN.

A potential shortcoming of using proposals as inputs to training ensemble members is a RPN likely finds many more examples of possible objects than actually are present. For a given image, an RPN may find hundreds of potential objects despite an image only containing a couple of true positive examples. This can be solved by setting the proposals with the highest confidence as the ground truth examples and labelling the remaining proposals as the background class. This is a stark comparison to the end-to-end training approach as there are now many more training examples and a large skew towards the background class. With this approach, the total number of training examples is increased from 80,116 ground truth object instances to 9,979,345 region proposals. The median of the almost 10 million proposals is 4,684 pixels, significantly smaller than the threshold of 19,205.5 determined using only ground truth boxes. This large increase in training examples and skew in data poses a question on how to split the RPN

proposals such that two networks can be trained towards small and large objects equally. As mentioned earlier, a pre-requisite in creating subsets of data is that the multiple sets should be roughly equal in regards to ground truth examples. If the subsets were split by the median of the RPN proposals (4,684) the two sets of data would have equal numbers of examples. However, the large skew in RPN proposals to smaller objects means that there significantly more ground truth samples in the subset of data containing larger objects. This can be seen in Table 2, where despite there being an almost even split in data subsets there are significantly more ground truth annotations in the RPN$_{larger}$ subset.

Table 2: Creating object resolution data subsets. If split by the median area of all region proposals training samples the larger dataset has significantly more ground truth object instance samples.

Data	**RPN**$_{small}$	**RPN**$_{larger}$
Ground Truth	19,992	60,116
Background	4,969,369	4,929,297
Total	4,989,361	4,989,413

Another option is to use the median of 19,205.5 found on only ground truth boxes. The data distribution based on this threshold can be seen in Table 3. In this instance there is significantly more data in the RPN$_{larger}$ subset, however, the skew is solely due to the many more background examples. The ground truth annotations are shared equally with 40,058 samples in each.

Table 3: Creating object resolution data subsets. If split by the median of area from ground truth objects there is an equal number of ground truth instances. However, RPN$_{larger}$ has significantly more background samples.

Data	**RPN**$_{small}$	**RPN**$_{larger}$
Ground Truth	40,058	40,058
Background	3,528,370	6,370,859
Total	3,568,428	6,410,917

As the overall goal of object detectors is to find objects within the classes, the decision was made to use the threshold of 19,205.5 to create the split in data. Despite there being significantly more background examples in one of the datasets. R-FCN ensemble members were trained on the two subsets of RPN according to the design guidelines. To evaluate how well the expert resolution members perform on the respective subsets of data tests were performed on splits of the 07 test data. This data was split by using the same median threshold of 19,205.5 used in creating the training subsets. Firstly, the results for small objects from 07 test can be seen in Table 4. Shown are R-FCNs trained on RPN$_{small}$, RPN$_{larger}$ and a baseline model trained on all 07+12 data. The shows that the model trained towards smaller object proposals on RPN$_{small}$ performs best. This trend is similarly true for large objects as seen in Table 5. Finally, for all ground truth objects the baseline model is the best performing as seen in Table 6.

Table 4: Results for R-FCN models trained on three different subsets of data and tested on only small objects from the 07 test set.

Train Data	**AP**
RPN$_{small}$	**55.00%**
RPN$_{larger}$	20.92%
07+12	43.80%

Table 5: Results for R-FCN models trained on three different subsets of data and tested on only large objects from the 07 test set.

Train Data	**AP**
RPN$_{small}$	21.28%
RPN$_{larger}$	**81.81%**
07+12	75.14%

Table 6: Results for R-FCN models trained on three different subsets of data and tested on all of the 07 test set.

Train Data	**AP**
RPN$_{small}$	46.74%
RPN$_{larger}$	62.48%
07+12	**79.59%**

Based upon these results it was determined that the ensemble member experts towards object size was suitable. The following section will explain the data sampling and training for image quality ensemble members.

4.4 IQA Detection

To evaluate the amount of distortions in the dataset a method for IQA is needed. A recent state of the art method is that of deep IQA (Bosse et al., 2016). Deep IQA is a CNN-based No-Reference (NR) IQA method that can be trained to measure the subjective visual quality of an image. It is deeper than previous CNN-based IQA methods with the architecture being inspired by VGG nets (Simonyan and Zisserman, 2015). Deep IQA consists of 14 convolutional layers, 5 max-pooling layers and 2 fully-connected layers. The architecture is shown in Figure 6. The convolutional layers are all 3×3 convolution kernels and activated using Rectified Linear Unit (ReLU). Inputs to each convolutional layer are zero-padded to

ensure output size is equal to the input. Max-pooling layers consist of 2×2 sized kernels. The network is trained on mini-batches of 32×32 patches. During inference non-overlapping patches are sampled from the image and image quality scores are predicted for each instance. The patch scores are averaged for the final score for the entire image.

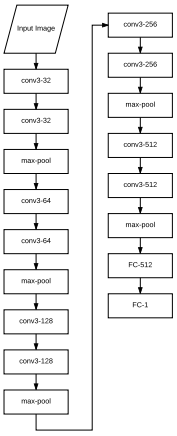


Figure 6: Architecture of the deep IQA network. Notation for convolutional layers are conv(receptive field size)-(number of channels) and fully-connected layers are FC(number of channels).

Training deep IQA requires a database of annotated images with both reference images and distorted counterparts. The following section will outline the database used in this project for IQA training.

4.5 LIVE Image Quality Database

Deep IQA assesses three different datasets for this purpose. These are LIVE which consists of 5 different distortions (Sheikh et al., 2006), TID2013 (Ponomarenko et al., 2013) with 24 different distortions and CSIQ (Larson and Chandler, 2009) with 5 types. For simplicity purposes the only LIVE dataset is chosen for this project. The dataset was made for the purposes of evaluating the subjective visual quality of images in regards to the five distortion types. The distortions are generated from 29 colour reference images that are of both high-resolution and high quality. An example of images from the dataset can be seen in Figure **??**. The references image were collected to have a wide variety in different content, this includes faces, people, animals, nature and man-made objects.

The five distortions generated from the reference images are Gaussian blur, white noise, JPEG compression, JPEG2K compression and fast fading. By varying the parameters used in creation of the distortions a larger database is created for each type. The total number of images is 982 where 174 are for Gaussian blur, white noise and fast fading. JPEG and JP2K compression have 233 and 227 images respectively. The distorted images were created as follows:

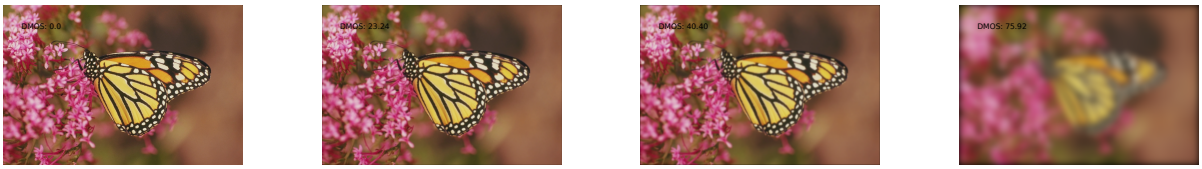
- Gaussian blur: blur is added to the images using a circular-symmetric Gaussian kernel of standard deviation σ_B. The values of σ_b are sampled between the range of 0.42 to 15 pixels.
- White noise: Gaussian white noise of standard deviation σ_N is added to all RGB pixels. Firstly, pixel values are scaled to between 0 and 1. σ_N varying between 0.012 and 2.0 is added, afterwards pixel values are rescaled back between 0 and 255.
- JPEG compression: compression artefacts are added to the reference bitmap images with JPEG at bit rates between 0.15 Bits per Pixel (BPP) to 3.34 BPP.
- JP2K compression: artefacts added ranging between 0.028 BPP to 3.15 BPP.
- Fast fading: this distortion represents errors that can occur when a JP2K bitstream is transmitted over a wireless channel. The receiver signal-to-noise-ratio is varied between 15.5 to 26.1dB for bit errors.

Subjective image quality values were calculated by showing human subjects all images, including reference images, and asking them to rate the image as either bad, poor, fair, good, or excellent. The rating was done using a slider on a graphical interface with the five possibilities being evenly spaced. A value between [1, 100] was then found depending on where the subject paced their rating. Difference Mean Opinion Score (DMOS) were calculated for each image and averaged between all users for the final image quality annotation for an image. A low DMOS represents high image quality and a high DMOS is a low quality image. Figure **??** shows an example of an image with four varying levels of Gaussian blur and their DMOS values.

Given the annotated DMOS values a system can be trained to predict the image quality of an image. The following section will outline how to train deep IQA for this purpose.

4.6 Training Deep IQA

In the original work the deep IQA model (Bosse et al., 2016) is trained for all five distortion types present in LIVE (Sheikh et al., 2006) (Sheikh et al.,). While this can provide insights into general image quality, individual models are needed to create a potentially



DMOS: 0.0 DMOS: 23.24 DMOS: 40.40 DMOS: 75.92

Figure 7: Four example images from the Gaussian blur distortion set. Respective DMOS scores are shown on the image and below.

more powerful ensemble. The NR-IQA model was provided by the authors and by taking advantage of the fine-tuning technique, models for each of the individual distortions can be trained in a timely manner. The model was provided for the Chainer framework (Tokui et al., 2015) and fine-tuning of individual models were done using this. Fine-tuning takes a previously trained model and uses these parameters as a starting point, rather than other commonly used initialisation techniques such as using a Gaussian distribution. As the model is already trained towards all distortions the assumption can be made that only a shorter training cycle is necessary to the new task. The five fine-tuned models are trained in a similar manner to that of the provided model following the guidelines in the original work (Bosse et al., 2016).

In the LIVE dataset there are 29 reference images from where the respective distortions have been created. When training deep IQA, the reference images are randomly split into 17 training images, 6 validation images and 6 test images. The deep IQA models are trained using mini-batches consisting of a total of 128 patches per forward/backward pass. The patches are sampled from four randomly selected images from the training split and each image accounts for 32 of the 128 patches in the mini-batch. This process is continued until no more patches are available for mini-batch sampling. This constitutes a completed epoch and all patches are again available for the next epoch. The model provided by the authors was trained for 3,000 epochs, however, as mentioned fine-tuning can drastically reduce the number of epochs required. Therefore, the models for the five distortions are trained for only 500 epochs each. The optimisation method for parameter updates is Adam (Kingma and Ba, 2014). The optimisation settings for Adam are unchanged to that of those used in training the original model. They are as $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$ and $\alpha = 10^{-4}$. A total of 10 models are trained for each distortion type, each on their individual random split of the 29 reference images. After each of the 500 epochs the model is evaluated on the validation set and the epoch with the best performance is chosen as the final model for testing. The evaluation metrics used for both the validation and test set is Pearson Linear Correlation Coefficient (LCC) and Spearman Rank Order Coefficient (SROCC). LCC is used for prediction accuracy as it is a measure of the linear correlation between two sets of data. SROCC evaluates the prediction monotonicity by measuring the rank correlation between the two sets. For both metrics a value of +1 indicates a positive correlation, 0 is no correlation, and -1 is a negative correlation.

The mean results for each of the distortion types from their 10 models can be seen in Table 7. Each best performing model on the respective validation sets are run on the testing sets and averaged. The results showed that well-performing deep IQA models were successfully trained towards the individual distortion types. The following section will cover how the PASCAL VOC data will be split in subsets based upon these deep IQA models.

Table 7: Average results from 10 trained deep IQA models for each distortion type.

Distortion Type	**LCC**	**SROCC**
Gaussian Blur	0.9750	0.9681
White Noise	0.9957	0.9887
JPEG	0.9805	0.9523
JP2K	0.9788	0.9600
FF	0.9679	0.9505

4.7 PASCAL VOC Data Split

Each model for the five distortion types and run through the 07+12 dataset in order to give an indication to the respective distributions, as done for the object sizes in Section **??** **??**. The distributions can been seen in the histograms in Figure 8.

The distribution for white noise and Gaussian blur is skewed towards a higher image quality as seen in Figure **??** and Figure **??** and also to a lesser extent in fast fading in Figure **??**. Whereas the image quality for compression distortions is somewhat of a Gaussian nature in Figure **??** and Figure **??**. For determining an appropriate manner to split the data the same constraints are made as in that for object sizes, namely that both subsets of data should have an equal number of ground truths to train on. Again by taking the me-

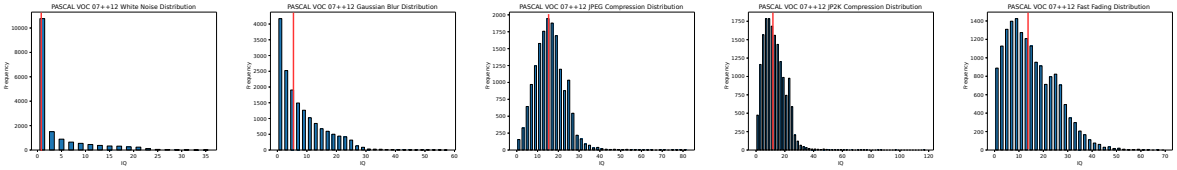


Figure 8: Histograms representing the distribution of image quality for the five distortions trained from the LIVE image quality dataset. The distortions shown are white noise (a), Gaussian blur (b), JPEG compression (c), JP2k compression (d), fast fading (e).

dian for each of the five distributions can satisfy this. The respective medians can be seen in Table 8.

Table 8: Threshold values used for each distortion type to create even subsets of training data from 07+12.

Distortion Type	Median
White Noise	0.599
Gaussian Blur	5.607
JPEG Compression	15.660
JP2K Compression	11.747
Fast Fading	13.373

A skewed distribution of data was also present for object sizes, however, creating two subsets of data for high and low white noise image quality does not appear to be feasible. The combination of both the heavy skew and half of the data lying below 0.599 indicates that a minimal amount of white noise distortion is present in the 07+12 dataset. Therefore, this distortion is not considered for part of the ensemble. While the Gaussian blur image quality is also skewed it is similar to that of the the object sizes and therefore is deemed appropriate to split based upon its median of 5.607. The remaining distributions are much less skewed and a total of eight R-FCN models will be trained for the high and low levels of image quality for the distortions Gaussian blur, JPEG compression, JP2K compression and fast fading. Therefore, in total there will be ten R-FCN models trained including the two for smaller and larger object sizes.

The 07+12 dataset has a total of 16,551 images and as this data is to be distributed into eight different subsets there is a possibility that there is a high level of overlap between the sets. As the aim of the ensemble is to learn to detect objects based upon different information, if subsets are two similar there may be potentially no advantage gained between two or more models. Therefore, a comparison matrix is used to evaluate how much the different combinations of 07+12 match. This can be seen for higher quality subsets in Table **??**, lower quality in Table **??** and between lower and higher quality in Table **??**.

[3]

4.8 Combining the Ensemble Members

A number of different strategies for combining the ensemble members will be described in this section. This includes averaging and weighted averaging the detections. The method for inferring each test image will be the same apart from the combination step. This is shown in Figure 9. For a given object proposal in an image found with the RPN each network will infer a bounding box and associated confidence for all classes. After this the given ensemble combination method determines the final detection.

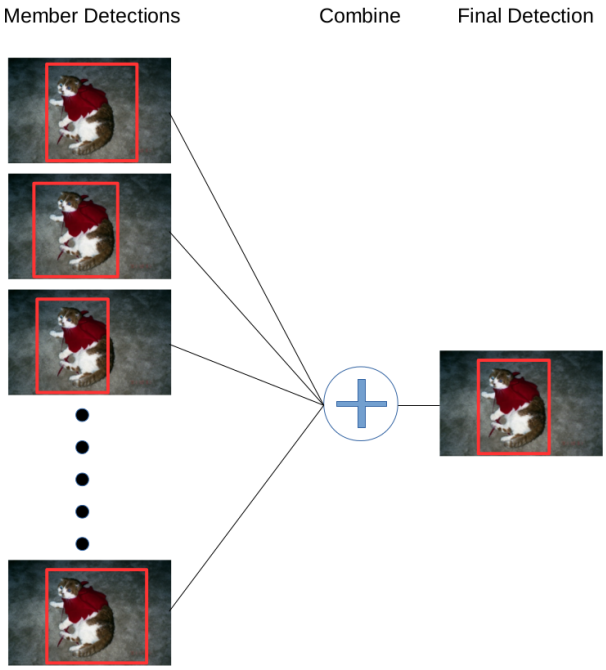


Figure 9: General overview for combining detections from different ensemble members. For a given proposal each member produces their respective detections. Then using a combination strategy they are combined for the final detection.

[3] FiXme Note: write info about overlap into text

A number of different combination strategies will be presented and evaluated in the remainder of this section.

4.9 Average Ensemble

One of the combination strategies is similar to that used when evaluating the expert member results earlier in this chapter. Each of the five ensemble factors are weighted evenly in the overall ensemble. Within each ensemble factor pair, the detection for one of the pairs will be chosen and the other discarded. This is determined by where the given factor lies for the given test image in relation to the training data distribution. For example, if for the given test image it is measured with a deep IQA to have JPEG compression below the threshold used to split the data, then the detection found using the model trained on that data will be used. This results in five detections that will be weighted equally to find the final detection by:

$$E_j = \frac{1}{n}\sum_{i=1}^{n} p_{i,j} \tag{1}$$

where n is the number of detections found by the n ensemble factor, p is the detection result to be averaged and i represents one of the ensemble factors. Finally, j is one of the five values found by each detection, namely the four corners of the bounding-box and the associated confidence.

4.10 Weighted Average Ensemble

Each of then 10 trained networks will be used on all object proposals found using the RPN. Weights will be distributed evenly across each of the five different types of factors as in the average ensemble. The weighted average ensemble is determined for each bounding-box and the associated confidence by:

$$E_j = \frac{1}{n}\sum_{i=1}^{n} w_i p_{i,j} \tag{2}$$

w_i is the weight for a given detection.Weights are determined in pairs for each of the 5 ensemble factors, where the total sum of weights is equal to n. If each detection were to be weighted equally all w would be equal to 1. As the weights are calculated in pairs each ensemble factor is overall weighted equally as the pair of weights can at most be equal to 2. By using this tactic in between the two sets of ensembles for a given factor can be weighted differently but overall each factor is weighted equally. Weights for a given factor is found according to where the the test image lies for that factors training data distribution. For example, the subsets of training data for Gaussian blur was determined according to the line shown in Figure 10.

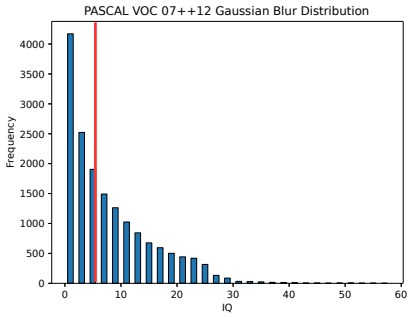


Figure 10: Distribution on the 07+12 train set for Gaussian blur found using a trained deep IQA.

The quality, q_i with respect to blur for a given image is determined using the appropriate deep IQA model, if the quality is below the value used to split the data the weights are calculated for the detection found with the given lower network by:

$$w_{Lower} = 2 - \frac{median_i - q_i}{median_i - minq_i} \tag{3}$$

and the weight for the upper network w_{Upper} by:

$$w_{Upper} = 2 - w_{Lower} \tag{4}$$

where $median_i$ is the value used to split the training data and $minq_i$ is the minimum quality for the given factor in the training set.

However, if the quality is above *split* the w_{Upper} is calculated by:

$$w_{Upper} = 2 - \frac{maxq_i - q_i}{maxq_i - median_i} \tag{5}$$

and lower weight w_{Lower}:

$$w_{Lower} = 2 - w_{Upper}. \tag{6}$$

It should also be noted that outliers are removed for the calculation of $minq_i$ and $maxq_i$ by removing the values below the 1% and above the 99% percentile. This ensures that the weighing of factors is not too heavily affected by large or small outlier values.

5 EXPERIMENTAL RESULTS

In this section the results for the two aforementioned ensemble combinations strategies will be presented. Each presentation will be accompanied with the result for the baseline R-FCN ResNet-101 model

trained on all of the 07+12 training data and will be dubbed as baseline. The results presented will be on the 2007 PASCAL VOC test set as also shown in earlier preliminary results in this report.

The results for both combination strategies can be seen in Table 9.

Table 9: Results for the two ensemble combination strategies and for the baseline model on the 2007 test set.

Method	**AP (%)**
Average	79.21%
Weighted Average	79.13%
Baseline	**79.59%**

While neither of the combinations provide an improvement over the baseline method both provide an increase in performance in comparison to the image quality expert results shown in Section **?? ??**. Here, individual members were 3-4% worse in performance in comparison to the baseline model on their trained expert areas. Additionally, the weighted average only performs slightly worse than that of the non-weighted version. This is interesting as the intra-factor experts for the image quality factors are similar in performance, however, while disregarding this and weighing models still provides a performance increase.

To the evaluate the contribution of both the eight quality factor ensemble members and the two resolution members these were combined separately based on the two strategies. The results for the average ensemble can be seen in Table 10 and the weighted ensemble in Table 11.

Table 10: Results for the the image quality ensemble members and resolution members individually combined using average strategy on the 2007 test set.

Ensemble Members	**AP (%)**
Image Quality	78.15%
Resolution	78.13%

Table 11: Results for the the image quality ensemble members and resolution members individually combined using the weighted average strategy on the 2007 test set.

Ensemble Members	**AP (%)**
Image Quality	78.44%
Resolution	75.00%

By separating the quality and resolution members Table 10 shows that the performance decreases by over 1% for both in comparison the the average ensemble result of 79.21%. This appears to indicate that the two complement each other well and have their own expertises for this problem. Table 11 also shows a decrease in performance when separating the members based on their expertise factors. The weighted average combination strategy does not show as large of a decrease in performance for only image quality as the average combination does, however, there is still a performance drop from 79.13% to 78.44%. There is a significant decrease in performance for the two resolution members showing an AP of 75.00% on the test set. This seems to show that the addition of weighing individual detections based on proposal size as a poorer approach. Comparing the two tables seems to indicate that image quality members are well suited to adding a weight to detection. Whereas, the resolution members are better suited to simply taking the detection from the appropriate model. Therefore, combinations of average and weighted average ensembles could be of interest. The results for these can be seen in Table 12. The two strategies are shown as either Image Quality or Resolution followed by the subscript $_{Avg}$ or $_{WAvg}$ indicating the combination strategies of average or weighted average respectively.

Table 12: Results for the the image quality ensemble members and resolution members with both combinations of average and weighted average on the 2007 test set.

Ensemble Members	**AP (%)**
Image Quality$_{WAvg}$ / Resolution$_{Avg}$	**79.90%**
Image Quality$_{Avg}$ / Resolution$_{WAvg}$	78.71%
Baseline	79.59%

Results in Table 12 show that by using separate strategies where image quality members are weighted and when resolution members are averaged only increases the performance. Additionally, the performance surpasses the baseline model. The increase is slight from 79.59% to 79.90%. However, again it appears that the members of the ensemble compliment each other well both intra-factor and inter-factor. As suspected the opposite strategy of average combination for image quality and weighted average for resolution does not surpass previous results.

The results so far have only been with different combinations of the expert ensemble members. However, another strategy is to include the baseline model trained on all of the 07+12 data. As the baseline model performs well by itself the other ensemble members will act as support, as ideally there are parts of the PASCAL VOC data that they perform better on due to the reduced training variance. The methods for ensemble are used as earlier, except that there is an additional member in the ensemble. Also it should be noted that as there is no complementary member to the baseline. Therefore, its detections are weighted by 1.0 regardless of ensemble combination strategy. Firstly, the results for the average and weighted average ensemble, both with the baseline model can be seen in Table 13. The inclusion of the baseline model

is shown by the subscript $_{base}$. The table shows that in both strategies the inclusion increases the overall performance. Using the weighted average the performance is increased by 0.22%. While the average strategy is increased above the baseline result by 0.65% to 79.86%.

Table 13: Results for the two ensemble combination strategies and for the baseline model on the 2007 test set. Shown is both the results with the expert ensemble members only and experts plus the baseline model.

Method	**AP (%)**
Average	79.21%
Average$_{base}$	**79.86%**
Weighted Average	79.13%
Weighted Average$_{base}$	79.35%
Baseline	79.59%

Next the addition of the baseline model with respective to each ensemble factor using the average ensemble strategy can be seen in Table 14. Both factors have a significant increase in AP performance with the extra ensemble member. The image quality experts gain 0.77%, while the two resolution members have their performance increased by 1.96%. The result of 80.09 is higher than the result shown in Table 12 even without having members trained towards image quality factors.

Table 14: Results for the the image quality ensemble members and resolution members individually combined using average strategy on the 2007 test set. Shown is both the results with the expert ensemble members only and experts plus the baseline model.

Ensemble Members	**AP (%)**
Image Quality	78.15%
Image Quality$_{base}$	78.92%
Resolution	78.13%
Resolution$_{base}$	**80.09%**
Baseline	79.59%

Adding the baseline model to the factors and using the weighted average strategy does not result in an improvement over the baseline result as shown in Table 15. However, both factors see a larger increase in performance than that of the average combination in Table 14. Image quality performance is increased by 0.71% and resolution members increase by 3.21%. Clearly regardless of ensemble strategy the addition of the baseline model aids in overall object detection on PASCAL VOC.

While improvements are seen for both strategies with the addition of baseline, the tendency is still that the resolution members perform best with the average ensemble and image quality with weighted average. Therefore, the two combinations of ensembles

Table 15: Results for the the image quality ensemble members and resolution members individually combined using weighted average strategy on the 2007 test set. Shown is both the results with the expert ensemble members only and experts plus the baseline model.

Ensemble Members	**AP (%)**
Image Quality	78.44%
Image Quality$_{base}$	79.15%
Resolution	75.00%
Resolution$_{base}$	78.21%
Baseline	**79.59%**

with the addition were tested. This is shown in Table 16 and shown is that this provided the best result of any ensemble combination. Image quality with the weighted average and resolution with average ensemble results in 80.15%, an increase of 0.56% in comparison to the baseline R-FCN.

Table 16: Results for the the image quality ensemble members and resolution members with both combinations of average and weighted average on the 2007 test set. Shown is both the results with the expert ensemble members only and experts plus the baseline model.

Ensemble Members	**AP (%)**
Image Quality$_{WAvg}$ / Resolution$_{Avg}$	79.90%
Image Quality$_{WAvg}$ / Resolution$_{Avg\ base}$	**80.15%**
Image Quality$_{Avg}$ / Resolution$_{WAvg}$	78.71%
Image Quality$_{Avg}$ / Resolution$_{WAvg\ base}$	79.10%
Baseline	79.59%

The AP results for each categories for best performing ensembles are shown in Table 17 and Table 18. The tables show results for the baseline model, the given ensemble method and the difference between the two for a given class. For the Resolution$_{base}$ ensemble, which had an overall increase of 0.5%, 13 of the 20 classes had an improvement compared to the baseline model. As seen in Table 17 the largest increases were for the TV class with 2.77%, train 2.53% and sofa 1.64%. The classes with the largest decrease in performance were horse falling by 2.21% and cow with 1.04%

Table 17: Results for the individual classes in the 2007 test set. Shown are the results for the baseline model and Resolution$_{base}$. Additionally the difference between the two methods are presented for a given class.

Model	**aero**	**bike**	**bird**	**boat**	**bottle**	**bus**	**car**	**cat**	**chair**	**cow**
Baseline	80.53	**84.59**	79.89	71.52	67.54	87.22	**87.59**	87.98	65.15	**87.11**
Resolution$_{base}$	**82.04**	84.21	**80.14**	**72.08**	**69.05**	**87.86**	87.56	**89.28**	**66.01**	86.07
Difference	+1.51	-0.58	+0.25	+0.56	+1.51	+0.64	-0.03	+1.30	+0.86	-1.04

Model	**table**	**dog**	**horse**	**mbike**	**person**	**plant**	**sheep**	**sofa**	**train**	**tv**
Baseline	**73.66**	88.61	**87.83**	83.21	79.87	**54.60**	**84.07**	80.03	83.60	77.17
Resolution$_{base}$	73.04	**89.10**	85.62	**83.59**	**79.95**	54.30	83.63	**81.67**	**86.13**	**80.47**
Difference	-0.62	+0.49	-2.21	+0.38	+0.08	-0.30	-0.44	+1.64	+2.53	+2.77

Table 18 shows that for the Image Quality$_{WAvg}$ / Resolution$_{Avgbase}$ ensemble again 13 of the classes in-

creased in performance. The largest increase were again train and TV with 2.77% and 2.48% respectively. Interestingly, the third largest winner with Resolution$_{base}$, sofa, decreased in performance in this instance. Giving an indication that object detection does not improve for all classes with the addition of image quality ensemble members. The worse performers in this instance were table and horse, decreasing by 1.63% and 0.87% respectively.

Table 18: Results for the individual classes in the 2007 test set. Shown are the results for the baseline model and Image Quality$_{WAvg}$ / Resolution$_{Avgbase}$. Additionally the difference between the two methods are presented for a given class

Model	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow
Baseline	80.53	84.59	79.89	71.52	67.54	87.22	**87.59**	87.98	65.15	**87.11**
Image Quality$_{WAvg}$ / Resolution$_{Avgbase}$	**81.41**	**85.79**	**81.09**	**72.87**	**69.09**	**88.00**	87.42	**89.12**	**66.71**	86.72
Difference	+0.88	+1.20	+1.20	+1.35	+1.55	+0.78	-0.17	+1.14	+1.56	-0.39

Model	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Baseline	**73.66**	88.61	**87.83**	83.21	79.87	**54.60**	**84.07**	**80.03**	83.60	77.17
Image Quality$_{WAvg}$ / Resolution$_{Avgbase}$	72.03	**88.69**	86.96	**84.24**	**80.09**	53.74	83.28	79.88	**86.28**	**79.65**
Difference	-1.63	+0.08	-0.87	+1.03	+0.22	-0.86	-0.79	-0.15	+2.68	+2.48

Finally, two examples of detections can be seen in Figure 12 and Figure **??**. For both figures, on the left is the full size image and right a zoomed version of the object and detections. The detections shown are for the ground truth annotation, baseline, Resolution$_{base}$ (Res) and Image Quality$_{WAvg}$ / Resolution$_{Avgbase}$ (IQ / Res). Additionally, shown in parentheses in the legend is the Intersection-Over-Union (IoU) between the ground truth and detection for the given method. Additional examples of detections can be seen in Section **?? ??**. For the boat detection in Figure 12, both ensemble methods increase the IoU significantly compared to the baseline detection. The IoU with the Resolution$_{base}$ is 0.045 higher at 0.845. The Image Quality$_{WAvg}$ / Resolution$_{Avgbase}$ detection matches even better with 0.061 higher intersection. The difference in the results seems to be largely due to the closer bounding box towards the right side of the boat, with the Image Quality$_{WAvg}$ / Resolution$_{Avgbase}$ fitting especially well.

Figure **??** is an example of a relatively small object in an image. In this instance the best fitting detection is from the Resolution$_{base}$ ensemble with an IoU of 0.825. An increase of 0.071 compared to the detection with the baseline model. The Image Quality$_{WAvg}$ / Resolution$_{Avgbase}$ also has a better fitting detection with an IoU of 0.803. The improvement for both ensembles seem to be towards to top of the train where both bounding boxes fit the ground truth better than the baseline detection. Again, both models had significant improvements for the train class compared to the baseline model.

6 DISCUSSION

7 CONCLUSION

ACKNOWLEDGEMENTS

If any, should be placed before the references section without numbering. To do so please use the following command: $\backslash section*\{ACKNOWLEDGEMENTS\}$

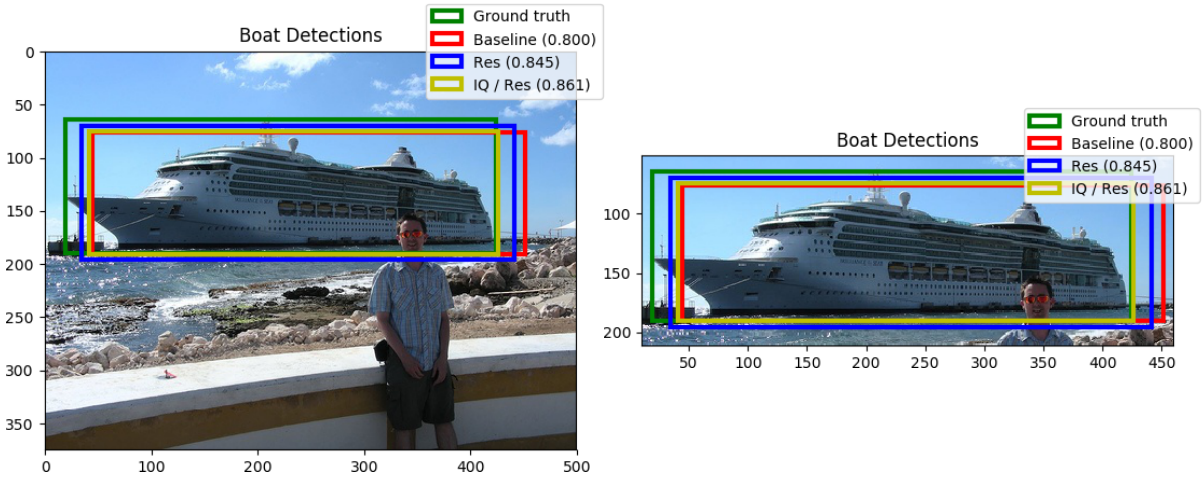


Figure 11: Detections for the boat class from an image in the 2007 test set. Shown are the bounding boxes for the ground truth annotation, baseline, Resolution$_{base}$ (Res) and Image Quality$_{WAvg}$ / Resolution$_{Avgbase}$ (IQ / Res). The IoU between the ground truth and bounding box is shown in parentheses for each method.

Figure 12: Detections for the boat class from an image in the 2007 test set. Shown are the bounding boxes for the ground truth annotation, baseline, Resolution$_{base}$ (Res) and Image Quality$_{WAvg}$ / Resolution$_{Avgbase}$ (IQ / Res). The IoU between the ground truth and bounding box is shown in parentheses for each method.

REFERENCES

Bosse, S., Maniry, D., Müller, K., Wiegand, T., and Samek, W. (2016). Deep neural networks for no-reference and full-reference image quality assessment. *CoRR*, abs/1612.01697.

COCO, M. (2017). Ms coco detections leaderboard.

Dai, J., Li, Y., He, K., and Sun, J. (2016). R-FCN: object detection via region-based fully convolutional networks. *CoRR*, abs/1605.06409.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338.

Girshick, R. (2015). Fast R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A.,

Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., and Murphy, K. (2016). Speed/accuracy trade-offs for modern convolutional object detectors. *CoRR*, abs/1611.10012.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc.

Larson, E. and Chandler, D. M. (2009). Consumer subjective image quality database.

Li, Y., Qi, H., Dai, J., Ji, X., and Wei, Y. (2016). Fully convolutional instance-aware semantic segmentation. *CoRR*, abs/1611.07709.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). *Microsoft COCO: Common Objects in Context*, pages 740–755. Springer International Publishing, Cham.

Long, J., Shelhamer, E., and Darrell, T. (2014). Fully convolutional networks for semantic segmentation. *CoRR*, abs/1411.4038.

Ponomarenko, N., Ieremeiev, O., Lukin, V., Egiazarian, K., Jin, L., Astola, J., Vozel, B., Chehdi, K., Carli, M., Battisti, F., and Kuo, C. C. J. (2013). Color image database tid2013: Peculiarities and preliminary results. In *European Workshop on Visual Information Processing (EUVIP)*, pages 106–111.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

Schroff, F. (2009). *Semantic Image Segmentation and Web-supervised Visual Learning*. University of Oxford.

Sheikh, H. R., Sabir, M. F., and Bovik, A. C. Live image quality assessment database release 2.

Sheikh, H. R., Sabir, M. F., and Bovik, A. C. (2006). A statistical evaluation of recent full reference image quality assessment algorithms. *IEEE Transactions on Image Processing*, 15(11):3440–3451.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *ICLR*.

Szegedy, C., Ioffe, S., and Vanhoucke, V. (2016). Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261.

Tokui, S., Oono, K., Hido, S., and Clayton, J. (2015). Chainer: a next-generation open source framework for deep learning. In *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS)*.

Uijlings, J. R. R., van de Sande, K. E. A., Gevers, T., and Smeulders, A. W. M. (2013). Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171.

Zhang, C. and Ma, Y. (2012). *Ensemble Machine Learning*. Springer US.

Zhang, X., Yang, Y.-H., Han, Z., Wang, H., and Gao, C. (2013). Object class detection: A survey. *ACM Comput. Surv.*, 46(1):10:1–10:53.

APPENDIX

If any, the appendix should appear directly after the references without numbering, and not on a new page. To do so please use the following command: $\backslash section*\{APPENDIX\}$