Analysis of Classification Performance Based on Cascade-Correlation and Auto-encoder

Zhe Xiong

Australian National University
u7150030@anu.edu.au

Abstract. Cascade-Correlation (CasCor) is one of the efficient and fast algorithms for supervised learning. Compared to a fully-connected feed-forward neural network, the topology of CasCor is unfixed and adaptive. CasCor starts with only input and output layers, and then adds the hidden neuron units one at a time as the training loss starts to be plateaued. With this way of training, CasCor can automatically determine the size of the neural network to match the complexity of a specific task. In this paper, we briefly introduce the CasCor architecture and algorithm, then implement the CasCor to handle a classification task on the Vehicle-X dataset. The auto-encoder is also be introduced for data compression. In the experimental part, we compare the performance of CasCor, CasCor + single-layer auto-encoder, CasCor + three-layer autoencoder, and fully connected network in the aspect of test accuracy and training time cost. The results show that the highest test accuracy was achieved by CasCor. The CasCor + single-layer auto-encoder costs the least amount of time to train.

Keywords: Cascade-Correlation · CasCor · Auto-encoder · Efficiency

1 Introduction

One of the key considerations in build artificial neural networks is the model complexity. There are some ways to decide the size of the neural network. The first way is to take the successful neural network architectures in another's study as a reference or use some rules of thumb. For example, the number of units in 1st hidden layers should larger than the number in the consequently hidden layers. The second way is to start with a large neural network, then reduce the size of the neural network by the pruning method [6]. The pruning method requires an inspection of the behavior of the hidden units by looking at the distinctiveness, relevance, contribution, and etc. The early stopping technique [3] can also achieve similar effects on the large neural networks. The third way is to build various architectures, then use cross-validation to compare the performances of those architectures [5]. The final architecture will be the model with the best performance. But the drawback of this way lies in the fact that the computational cost usually very high [5]. The fourth way to govern the model complexity is to introduce the regularization term to the model. Regularization terms can prevent the model from over-fitting and enhance the generalization of the model. The Bayesian method works similarly to the regularization term, which implicitly regularizes the parameters in the model.

In this paper, a constructive architecture and algorithm are used to dealing with the problem of the model complexity. Cascade-Correlation (CasCor) [2] begins with a minimal network, and then incrementally builds a neural net by adding new hidden units one by one during the training process, leading to a multi-layer structure that matches the complexity of the specific task. In the way, for the CasCor, there is no need to determine the number of hidden units ahead of the training process.

2 Methodology

2.1 Cascade-Correlation

The Cascade-Correlation architecture starts with minimal size, only a fully connected input layer and output layer without hidden layer or units as shown in Initial in Figure 1. The size of the input and output layer depends on the input dimension and the number of classes of the specific task.

In Figure 1, stage 1 and stage 2 show how the hidden units are added to the main architecture. The cascade architecture implies that the hidden units are added one at a time under a certain criterion. Specifically, the learning Algorithm starts with training the neural network only with the input and output layers with a usual learning algorithm. At some point, the training will reach the plateau, the error cannot be further decreased. If further error decrement is required, a new hidden neuron needs to be inset into the network. At this stage, candidate units are generated. The candidate units are connected to the input units and all the existing hidden units, but there is no connection between the candidate unit and the output units. Next,

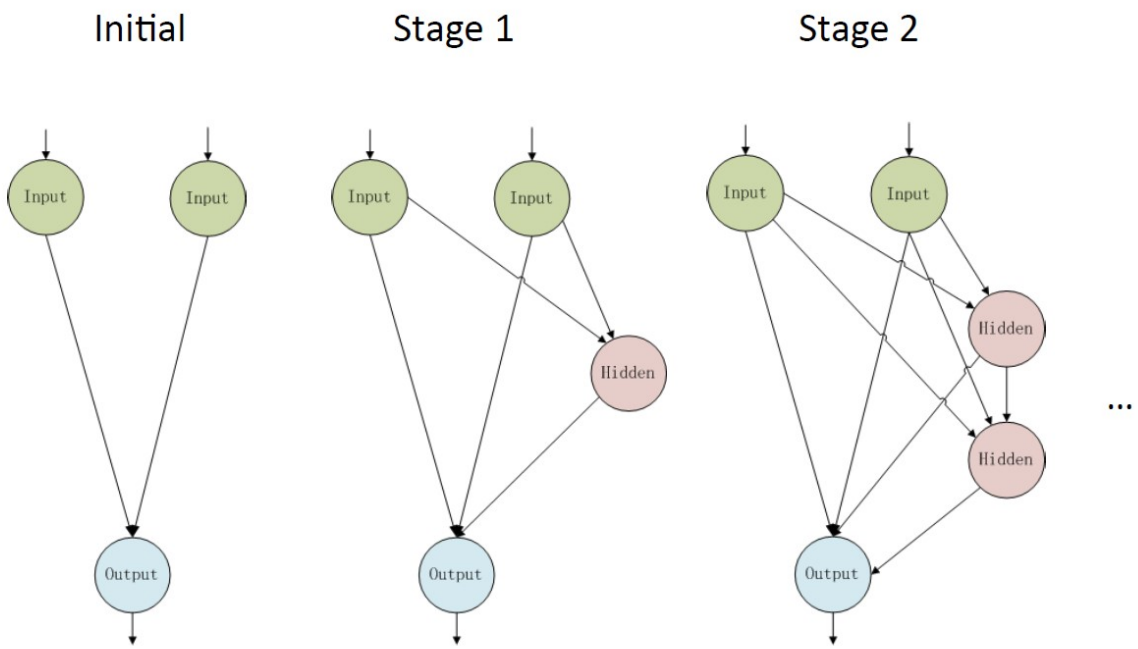

Fig. 1. Illustration of the CasCor topology.

CasCor maximizes the correlation score between the activation output of the candidate unit and the residual loss to train all the links connected to the candidate unit. When the correlation score is stopping to increase, the candidate unit with the highest correlation is added to the networks with the frozen weights and the links between the selected unit and all the output units are generated. As shown in Figure 2, all the links trained for the maximization of the correlation score are frozen (shown in boxed connection in Figure 2), but the links between candidate units and output units are still alive (shown in crossed connection in Figure 2). Last, CasCor continues to train the new architecture with the usual learning algorithm and repeats adding new hidden units until the training loss reaching the desired level.

2.2 Experimental Dataset

The experimental dataset is synthesized by Vehicle-X [8]. In this paper, the input data is the feature vectors extracted from the ResNet model which is pre-trained on ImageNet. For the experimental dataset, there are 45,438 samples for the training data, 14,936 samples for the validation data, and 15,142 samples for the testing data. In all three sets, each sample contains 2048 features, and 10 types of labels including vehicle id, vehicle orientation, light intensity, light direction, camera distance, camera height, camera id, vehicle type, color, and image name. In this paper, only the vehicle id is used in the classification task. There is 1,362 vehicle id in the dataset, therefore the total number of the class is 1,362 in this task.

2.3 Auto-encoder

An auto-encoder is a type of artificial neural network used to learn efficient data codings in an unsupervised manner[4]. The purpose of the encoder is to extract the key representation from the input. The basic architecture of the auto-encoder is shown in Figure 3. It is a feed-forward, non-recurrent neural network. There is an encoder between the input layer and ”bottleneck” hidden layer, and a decoder ”bottleneck” hidden layer and output layer. The role of the encoder is for compressing the data, and the role of the decoder is for reconstructed the data. In the task of data compression, the input layer and output layer are identical, therefore, the goal of the whole network is to minimize the reconstruction error. After a certain number of epochs, the compressed data is represented in the ”bottleneck” hidden layer.

In this paper, The auto-encoders are used to pre-train the dataset for further complexity deduction. a single-layer auto-encoder and a three-layer auto-encoder are used to compressed data from 2049-dimension to 512-dimension.

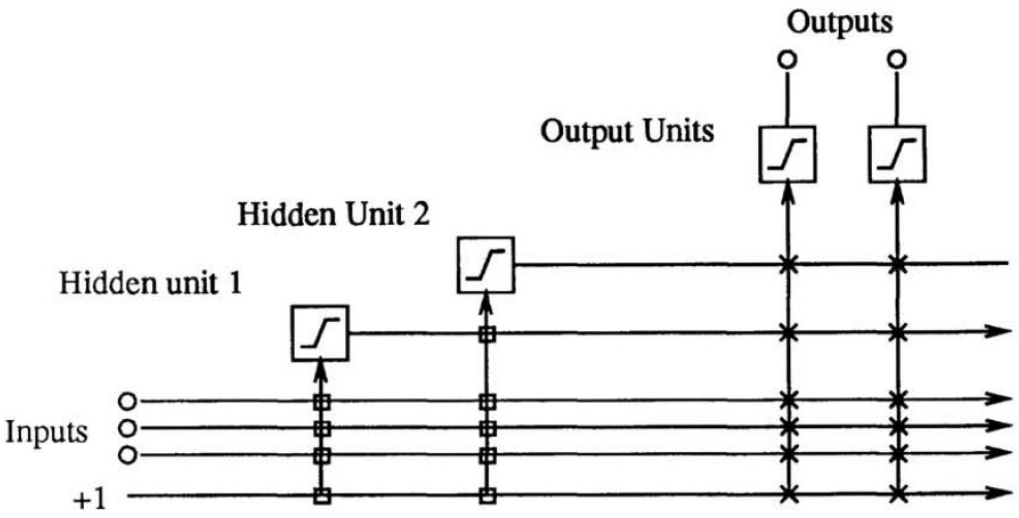

Fig. 2. Illustration of adding new hidden units for the CasCor architecture [2]

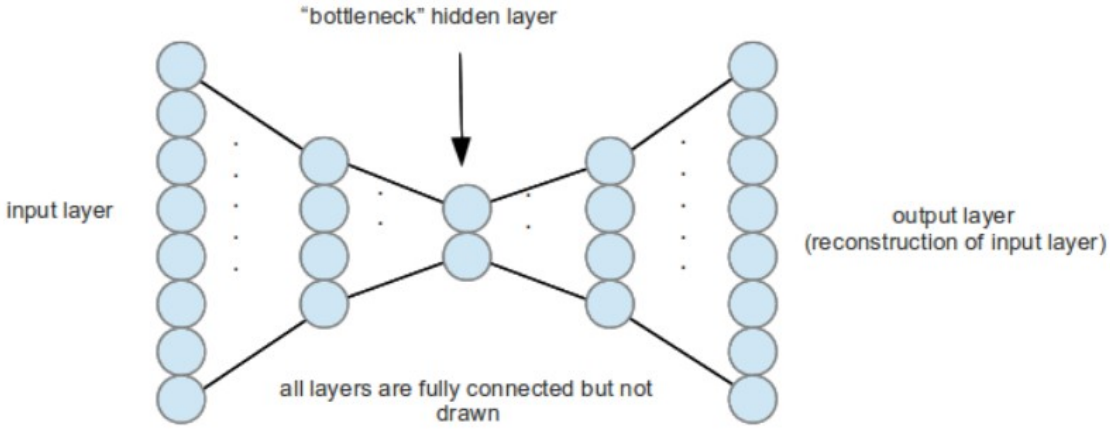

Fig. 3. Auto-encoder Architecture

2.4 Experimental Procedure

We first introduce the data loading for the Vehicle-X dataset and then provide a detailed description of the training scheme of the CasCor.

a) Data Loading. For the data loading of the Vehicle-X dataset, the 2048-dimensional feature vector of each sample is saved in the *.npy* file and the corresponding class name is the first 5 characters of the *.npy* filename. We encode the class name to the label number range from 0 to 1361. To apply the bias item of the Cascade-Correlation, for each sample, we expand the feature vector to a 2049-dimensional vector by adding a constant term (*i.e*, 1) for the input data.

b) Data Compression. For data compression, there are two auto-encoders. The First auto-encoder has a single hidden layer. It compresses the dataset from 2049-dimensional to 512-dimensional. The second auto-encoder has three hidden layers. It compresses the dataset from 2049-dimensional to 1024-dimensional, and then to 512-dimension. Training for both auto-encoders has 100 epochs and with batch size equal to 128.

c) Training Scheme of Cascade-Correlation. For the Cascade-Correlation, we apply a stochastic gradient descent (SGD) optimizer [7] with the learning rate 0.01 for the usual learning algorithm to minimize the training loss of the model, where we use cross-entropy loss [1] as the training loss, set the minimization epoch of each round as 10, and set the batch size as 128. For all the hidden units, we use the *sigmoid* function as the activation function. For the correlation score, we use the correlation between the activation of the new hidden unit and the residual cross-entropy loss of the loss. To maximize the correlation score, we apply a gradient descent optimizer on all the training data points with the learning rate of 0.01, where we set the maximization step as 30. We set the maximum number of the hidden units as 200.

3 Results and Discussions

a) Classification accuracy. We first report the training accuracy and the test accuracy of the CasCor model with the increasing number of hidden units. As shown in Figure 4, we can see that, during the training, the training and test accuracies increase with the increasing number of the hidden units. When the number of the hidden units is larger than 120, the test accuracy stops to increase but the training accuracy continues to increase, which indicates the CasCor model starts to overfit on the training data when the number of the hidden units is larger than 120.

b) Cross-entropy loss. Correspondingly, we report the average training loss and the average test loss of the CasCor model with the increasing number of the hidden units during the training. As shown in Figure 5, we can see that, When the number of the hidden units is larger than 120, the average test loss starts to converge but the average training loss continues to decrease. It also indicates the CasCor model is easy to overfit on the training data when the number of the hidden unit is large.

c) Compared CasCor with the fully-connected network. We compare the CasCor with the fully-connected network, where all the weights of the fully-connected network are trainable. For the fully-connected network, we set the number of the hidden layers as 1, the number of the hidden units as 200, the batch size as 128, the training epoch as 1000. As shown in Table 1, we report the test accuracy of the CasCor and the fully-connected network. We can see that the CasCor achieves higher test accuracy than the fully-connected network. For this reason, on one hand, the architecture of the CasCor is adaptive and can be deeper than the fully-connected network with one hidden layer. On the other hand, the weights of the hidden units for the CasCor are frozen during the minimization of the training loss, which alleviates the overfitting problem to a certain extent.

d) Compared compressed data with original data We compare the CasCor with orignal data as input and compressed data as input. For compressed data, there are two auto-encoders. in Table 1, we report the test accuracy for CasCor on compressed data from a single-layer auto-encoder and a three-layer auto-encoder. The results show that the test accuracy for CasCor + single-layer auto-encoder are slightly higher than the test accuracy for CasCor + three-layer auto-encoder. Both compressed datasets lead test accuracy reduction compared to CasCor with the original dataset.

e) Training time cost. Since the weights of the hidden units for the CasCor are frozen during the minimization of the training loss, the cost of the back-propagation to update the model weights is less than the original back-propagation. We compare the training time cost for the CasCor under the back-propagation of the CasCor and the original back-propagation when the number of the hidden units are 200 (*i.e.*, when the architecture of the CasCor is fixed.). The experiment are conducted on a single NVIDIA TITAN X GPU. As shown in Table 2, we report the training time cost of 5 epochs. As we can see, the back-propagation of the CasCor can lead to less training time cost than the original back-propagation. For CasCor with compressed data, the training time is further reduced. Although the dimension of data is significantly reduced from 2049 to 512, the training time is only reduced slightly. The training time cost is dominated by the CasCor algorithm.

Table 1. Test accuracy (%) of the CasCor and the fully-connected network.

Model	Test accuracy
fully-connected network	5.85
CasCor	11.85
CasCor + single-layer auto-encoder	10.21
CasCor + three-layer auto-encoder	10.06

Table 2. Training time cost under the back-propagation of the CasCor and the original.

Method	Training time cost (5 epochs)
Original	149.73s
CasCor	37.56s
CasCor + single-layer auto-encoder	35.24s
CasCor + three-layer auto-encoder	35.32s

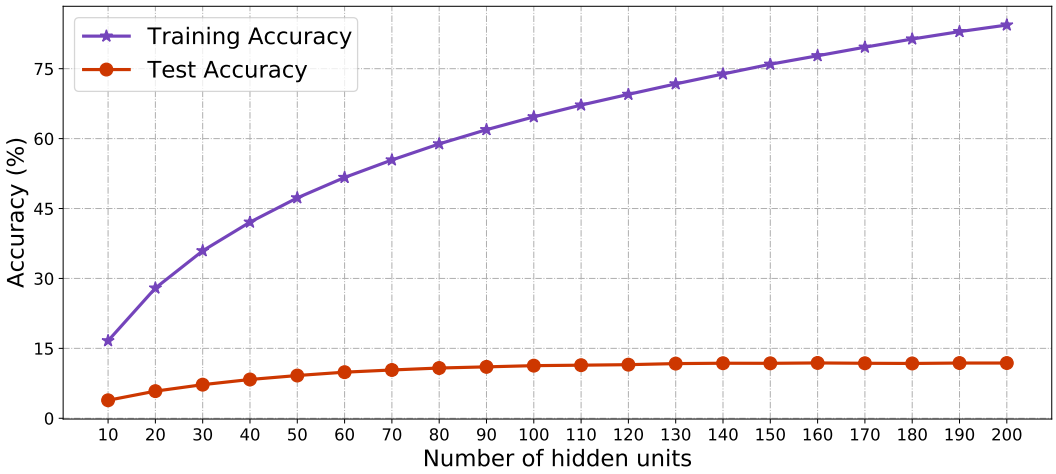

Fig. 4. Training accuracy and test accuracy of the CasCor model with the increasing number of the hidden units during the training.

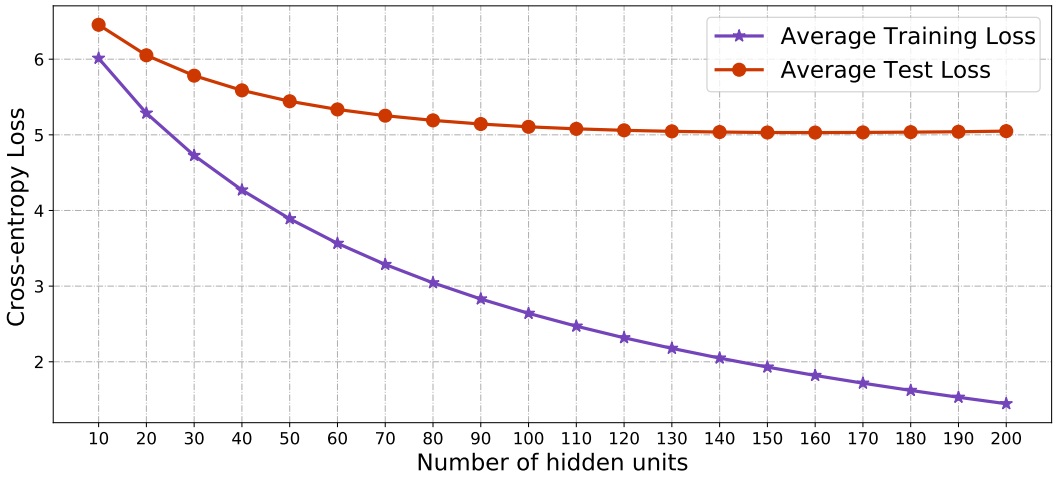

Fig. 5. Average training loss and test loss of the CasCor model with the increasing number of the hidden units during the training.

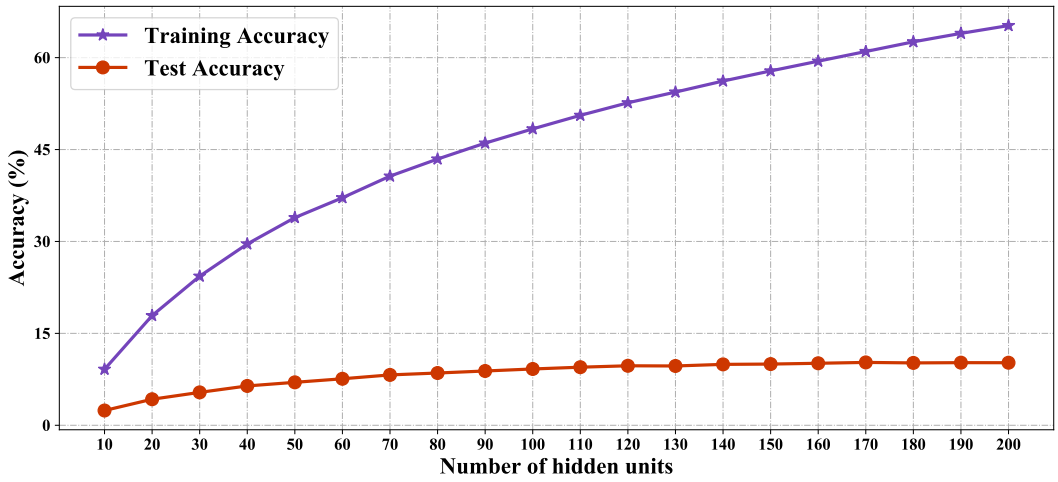

Fig. 6. Training accuracy and test accuracy of the CasCor model + single-layer auto-encoder with the increasing number of the hidden units during the training.

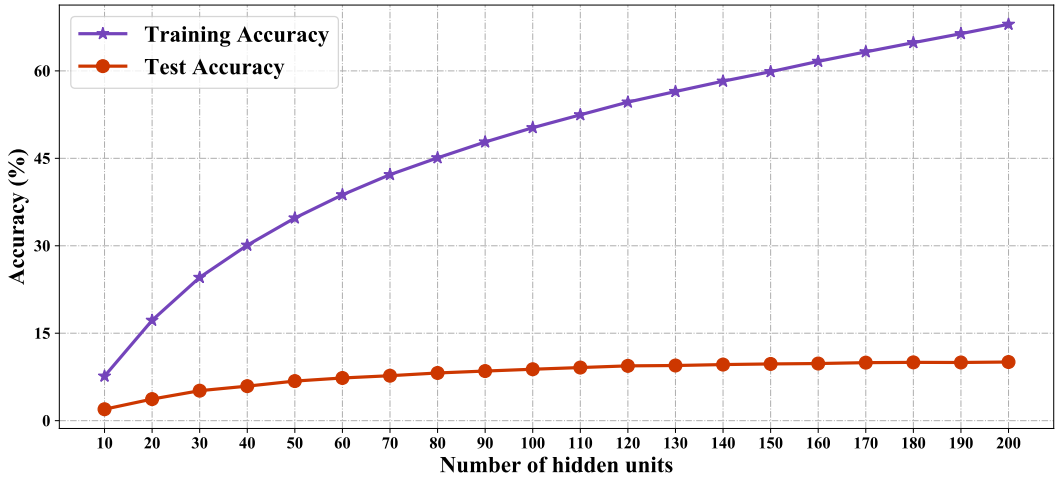

Fig. 7. Training accuracy and test accuracy of the CasCor model + three-layer auto-encoder with the increasing number of the hidden units during the training.

4 Conclusion and future work

The main purpose of this paper is to analyze the performance of the Cascade-Correlation Neural Network. The datasets for experimentation are sets of pre-trained feature vectors extracted from Vehicle-X. By compare performance on the test accuracy and training time cost in both Cascade-Correlation network (CasCor) and fully-connected network, we found that the test accuracy almost doubled in CasCor, meanwhile, the training time for CasCor is less than one-fifth of the training time for fully-connected network. The rationale behind the result is that CasCor can determine the number of hidden units according to the complexity of the task. Therefore, the final hidden units in CasCor are less than the number of hidden units in fully-connected network. The fewer hidden units reduce the training time in CasCor and increase the generality by preventing over-fitting.

A future study can focus on the CasCor network with more cascade layers and comparing it to multi-layer fully-connected network. Other related architecture variations, for example, Casper, Local Feature Constructive Cascade (LoCC) and Symmetry Local Feature Constructive Cascade (SymLoCC) are also worth exploring. In this paper, we only conduct experiments on

classification tasks and image data. In the future, experiments on regression and sequential data are one possible way to move on.

References

1. Bishop, C.M.: Pattern recognition and machine learning. springer (2006)
2. Fahlman, S.E., Lebiere, C.: The cascade-correlation learning architecture. Tech. rep., CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE (1990)
3. Finnoff, W., Hergert, F., Zimmermann, H.G.: Improving model selection by nonconvergent methods. Neural Networks **6**(6), 771–783 (1993)
4. Kramer, M.A.: Nonlinear principal component analysis using autoassociative neural networks. AIChE journal **37**(2), 233–243 (1991)
5. Moody, J.: Prediction risk and architecture selection for neural networks. In: From statistics to neural networks, pp. 147–165. Springer (1994)
6. Reed, R.: Pruning algorithms-a survey. IEEE transactions on Neural Networks **4**(5), 740–747 (1993)
7. Robbins, H., Monro, S.: A stochastic approximation method. The annals of mathematical statistics pp. 400–407 (1951)
8. Yao, Y., Zheng, L., Yang, X., Naphade, M., Gedeon, T.: Simulating content consistent vehicle datasets with attribute descent. In: ECCV (2020)