Learning to Detect Roads in High-Resolution Aerial Images

Volodymyr Mnih and Geoffrey E. Hinton

Department of Computer Science, University of Toronto,
6 King's College Rd., Toronto, Ontario,
M5S 3G4, Canada
{vmnih,hinton}@cs.toronto.edu

Abstract. Reliably extracting information from aerial imagery is a difficult problem with many practical applications. One specific case of this problem is the task of automatically detecting roads. This task is a difficult vision problem because of occlusions, shadows, and a wide variety of non-road objects. Despite 30 years of work on automatic road detection, no automatic or semi-automatic road detection system is currently on the market and no published method has been shown to work reliably on large datasets of urban imagery. We propose detecting roads using a neural network with millions of trainable weights which looks at a much larger context than was used in previous attempts at learning the task. The network is trained on massive amounts of data using a consumer GPU. We demonstrate that predictive performance can be substantially improved by initializing the feature detectors using recently developed unsupervised learning methods as well as by taking advantage of the local spatial coherence of the output labels. We show that our method works reliably on two challenging urban datasets that are an order of magnitude larger than what was used to evaluate previous approaches.

1 Introduction

Having up-to-date road maps is crucial for providing many important services. For example, a city requires accurate road maps for routing emergency vehicles, while a GPS-based navigation system needs the same information in order to provide the best directions to its users. Since new roads are constructed frequently keeping road maps up-to-date is an important problem.

At present, road maps are constructed and updated by hand based on high-resolution aerial imagery. Since very large areas need to be considered, the updating process is costly and time consuming. For this reason automatic detection of roads in high-resolution aerial imagery has attracted a lot of attention in the remote sensing community. Nevertheless, despite over 30 years of effort [1], at the time of writing there was no commercial automatic or semi-automatic road detection system on the market [2, 3] and, to the best of our knowledge, no published method has been shown to work reliably on large datasets of high-resolution urban imagery.

Much of the published work on automatic road detection follows an ad-hoc multi-stage approach [1, 4, 5]. This generally involves establishing some a priori criteria for the appearance of roads and engineering a system that detects objects that satisfy the established criteria. For example, roads are often characterized as high-contrast regions with low curvature and constant width, with a typical detection strategy involving edge detection, followed by edge grouping and pruning. While some of these approaches have exhibited good performance on a few sample images, the way in which they combine multiple components often results in the need to tune multiple thresholds and such methods have not been shown to work on large real-world datasets.

In this paper we follow a different approach, where the system *learns* to detect roads from expert-labelled data. Learning approaches are particularly well-suited to the road detection task because it is a rare example of a problem where expert-labelled data is abundant. It is easy to obtain hundreds of square kilometers of high-resolution aerial images and aligned road maps. In fact, most universities have libraries dedicated solely to geographic data of this kind.

Learning-based approaches to road detection are not new – several attempts at predicting whether a given pixel is road or not road given features extracted from some context around it have been made [6–9]. While showing some promise, these approaches have also failed to scale up to large challenging datasets. We believe that previous learning-based approaches to road detection have not worked well because they suffer from three main problems. First, very little training data is used, likely because ground truth for training and testing is typically obtained by manually labelling each pixel of an aerial image as road or non-road making it infeasible to use a lot of training data. Second, either a very small context is used to extract the features, or only a few features are extracted from the context. Finally, predictions for each pixel are made independently, ignoring the strong dependencies between the road/non-road labels for nearby pixels.

We propose a large-scale learning approach to road detection that addresses all three problems as follows:

- We use synthetic road/non-road labels that we generate from readily available vector road maps. This allows us to generate much larger labelled datasets than the ones that have been used in the past.[1]
- By using neural networks implemented on a graphics processor as our predictors we are able to efficiently learn a large number of features and use a large context for making predictions.
- We introduce a post-processing procedure that uses the dependencies present in nearby map pixels to significantly improve the predictions of our neural network.

Our proposed approach is the first to be shown to work well on large amounts of such challenging data. In fact, we perform an evaluation on two challenging urban datasets covering an area that is an order of magnitude larger than what was used to evaluate any previous approach. We also show that a previous learning based approach works well on some parts of the datasets but very poorly on others. Finally, we show that all three of our proposed enhancements are important to obtaining good detection results.

[1] Dollar et al. [10] proposed a similar approach to generating ground truth data but still used very little training data.

2 Problem Formulation

Let S be a satellite/aerial image and let M be a corresponding road map image. We define $M(i,j)$ to be 1 whenever location (i,j) in the satellite image S corresponds to a road pixel and 0 otherwise. The goal of this paper is to learn $p(M(i,j)|S)$ from data.

In a high-resolution aerial image, a single pixel can represent a square patch of land that is anywhere between several meters and tens of centimeters wide. At the same time one is typically interested in detecting roads in a large area such as an entire town or city. Hence, one is generally faced with the problem of making predictions for millions if not billions of map pixels based on an equally large number of satellite image pixels. For these reasons, the probability that $M(i,j) = 1$ has typically been modeled as a function of some relatively small subset of S that contains location (i,j) instead of the entire image S [7, 10]. In this paper we model

$$p(N(M(i,j), w_m)|N(S(i,j), w_s)), \tag{1}$$

where $N(I(i,j), w)$ denotes a $w \times w$ patch of image I centered at location (i,j). Hence, we learn to make predictions for a $w_m \times w_m$ map patch given a $w_s \times w_s$ satellite image patch centered at the same location, where $w_m < w_s$. This allows us to reduce the required computation by both limiting the context used to make the predictions and by reusing the computations performed to extract features from the context.

2.1 Data

While high-resolution aerial imagery is easy to obtain, per pixel road/non-road labels are generally not available because most road maps come in a vector format that only specifies the centreline of each road and provides no information about road widths. This means that in order to obtain per-pixel labels one must either label images by hand or generate approximate labels from vector data. The hand labelling approach results in the most accurate labels, but is tedious and expensive. In this paper we concentrate on using approximate labels.

Our procedure for generating per-pixel labels for a given satellite image S is as follows. We start with a vector road map consisting of road centreline locations for a region that includes the area depicted in S. We rasterize the road map to obtain a mask C for the satellite image S. In other words, $C(i,j)$ is 1 if location (i,j) in satellite image S belongs to a road centreline and 0 otherwise.

We then use the mask C to define the ground truth map M as

$$M(i,j) = e^{-\frac{d(i,j)^2}{\sigma^2}}, \tag{2}$$

where $d(i,j)$ is the Euclidean distance between location (i,j) and the nearest nonzero pixel in the mask C, and σ is a smoothing parameter that depends on the scale of the aerial images being used. $M(i,j)$ can be interpreted as the probability that location (i,j) belongs to a road given that it is $d(i,j)$ pixels away from the nearest centreline pixel. This soft weighting scheme accounts for uncertainty in road widths and centreline locations. In our experiment σ was set such that the distance equivalent to $2\sigma + 1$ pixels roughly corresponds to the width of a typical two-lane road.

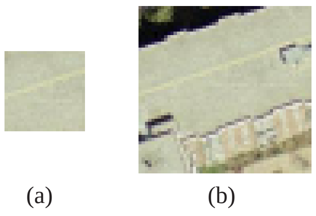


Fig. 1. The rooftop of an apartment building. a) Without context. b) With context.

3 Learning to Detect Roads

Our goal is to learn a model of (1) from data. We use neural networks because of their ability to scale to massive amounts of data as well as the ease with which they can be implemented on parallel hardware such as a GPU. We model (1) as

$$f(\phi(N(S(i,j),w_s))), \tag{3}$$

where ϕ is feature extractor/pre-processor and f is a neural network with a single hidden layer and logistic sigmoid hidden and output units. To be precise,

$$f(\mathbf{x}) = \sigma(\mathbf{W}_2^T\sigma(\mathbf{W}_1^T\mathbf{x}+\mathbf{b}_1)+\mathbf{b}_2), \tag{4}$$

where $\sigma(\mathbf{x})$ is the elementwise logistic sigmoid function, $\mathbf{W}$'s are weight matrices and $\mathbf{b}$'s are bias vectors. We now describe the pre-processing function ϕ, followed by the training procedure for f.

3.1 Pre-processing

It has been pointed out that it is insufficient to use only local image intensity information for detecting roads [7]. We illustrate this point with Figure 1. The aerial image patch depicted in sub-figure 1(a) resembles a patch of road, but with more context, as shown in sub-figure 1(b), it is clearly the roof of an apartment building. Hence, it is important to incorporate as much context as possible into the inputs to the predictor.

The primary aim of the pre-processing procedure is to reduce the dimensionality of the input data in order to allow the use of a large context for making predictions. We apply Principal Component Analysis to $w_s \times w_s$ RGB aerial image patches and retain the top $w_s \cdot w_s$ principal components. The function ϕ is then defined as the projection of $w_s \times w_s$ RGB image patches onto the top $w_s \cdot w_s$ principal components. This transformation reduces the dimensionality of the data by two thirds while retaining most of the important structure. We have experimented with using alternative colour spaces, such as HSV, but did not find a substantial difference in performance.

It is possible to augment the input representation with other features, such as edge or texture features, but we do not do so in this paper. We have experimented with using edge information in addition to image intensity information, but this did not improve

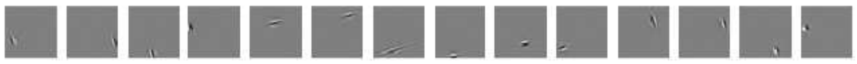


Fig. 2. Some of the filters learned by the unsupervised pretraining procedure.

performance. This is likely due to our use of an unsupervised learning procedure for initializing, or pretraining, the neural network. In the next section we will describe how this procedure discovers edge features independently by learning a model of aerial image patches.

3.2 Training Procedure

At training time we are presented with N map and aerial image patch pairs. Let $\mathbf{m}^{(n)}$ and $\mathbf{s}^{(n)}$ be vectors representing the nth map and aerial image patches respectively, and let $\hat{\mathbf{m}}^{(n)}$ denote the predicted map patch for the nth training case. We train the neural network by minimizing the total cross entropy between ground truth and predicted map patches given by

$$-\sum_{n=1}^{N}\sum_{i=1}^{w_m^2}\left(m_i^{(n)}\log\hat{m}_i^{(n)} + (1-m_i^{(n)})\log(1-\hat{m}_i^{(n)})\right), \tag{5}$$

where we use subscripts to index vector components. We used stochastic gradient descent with momentum as the optimizer.

Unsupervised Pretraining Traditionally neural networks have been initialized with small random weights. However, it has recently been shown that using an unsupervised learning procedure to initialize the weights can significantly improve the performance of neural networks [11, 12]. Using such an initialization procedure has been referred to as *pretraining*.

We pretrain the neural network f using the procedure of Hinton and Salakhutdinov [11], which makes use of Restricted Boltzmann Machines (RBMs). An RBM is a type of undirected graphical model that defines a joint probability distribution over a vector of observed variables $\mathbf{v}$ and a vector of latent variables $\mathbf{h}$. Since our neural network has real-valued inputs and logistic hidden units, in order to apply RBM-based pretraining, we use an RBM with Gaussian visible and binary hidden units. The joint probability distribution over $\mathbf{v}$ and $\mathbf{h}$ defined by an RBM with Gaussian visible and binary hidden units is

$$p(\mathbf{v},\mathbf{h}) = e^{-E(\mathbf{v},\mathbf{h})}/Z,$$

where Z is a normalizing constant and the energy $E(\mathbf{v},\mathbf{h})$ is defined as

$$E(\mathbf{v},\mathbf{h}) = \sum_i v_i^2 - \left(\sum_i c_i v_i + \sum_k b_k h_k + \sum_{i,k} w_{ik} v_i h_k\right). \tag{6}$$

While maximum likelihood learning in RBMs is generally intractable, efficient approximate learning can be performed by approximately minimizing a different objective function known as Contrastive Divergence [13].

We train an RBM on the PCA representations of aerial image patches by approximately minimizing Contrastive Divergence using stochastic gradient descent with momentum. In order to encourage a sparse model of aerial images, i.e. one where only a few components of **h** are nonzero, we fix the hidden unit biases b_k to a large negative value[2], as proposed by Norouzi et al. [14]. This encourages the hidden units to be off unless they get a large input from the visible units. Once the RBM was trained, we initialized the weight matrix W_1 and bias vector b_1 from Equation 4 with the RBM weights w and b. We found that encouraging sparseness sped up learning and improved generalization.

Some selected filters learned by the pretraining procedure are shown in Figure 2. The vast majority of the filters learned to ignore colour, but the few filters that were colour sensitive were low-frequency, opposing red-green or blue-yellow filters. Many of the colour-neutral filters are oriented, high-frequency edge filters. We believe this is why augmenting the inputs with edge information did not improve road detection performance.

Adding Rotations When training the neural network f we found that it is useful to rotate each training case by a random angle each time it is processed. Since many cities have large areas where the road network forms a grid, training on data without rotations will result in a model that is better at detecting roads at certain orientations. By randomly rotating the training cases the resulting models do not favor roads in any particular orientation.

4 Incorporating Structure

Figure 3(a) shows predictions for a small map patch made by our neural network. There are two obvious problems with these predictions – there are both gaps in the predicted roads and disconnected blotches of road pixels. Given our prior knowledge about the structure of road networks it would be safe to conclude that the blotches in Figure 3(a) are false positives while the gaps are false negatives. Previous learning-based approaches to road detection along with the method described in Section 3 make such mistakes because they make predictions independently for all pixels.

In order to take advantage of the structure present in nearby road/non-road labels we introduce a post-processing step. The goal is to improve the prediction for a given map pixel using nearby predictions. We treat this as a supervised learning problem and train a neural network to predict a $w_m \times w_m$ map patch from a $w_c \times w_c$ patch of predictions. To be precise, let $\hat{M}$ be the predictions of neural network f for map image M. Then let f_p be a neural network of the same functional form as f that predicts $N(M(i,j), w_m)$ based on $N(\hat{M}(i,j), w_c)$. The prediction of f_p for map image M is then denoted by $\hat{M}_p$.

[2] In this paper, we set b_k to -4.

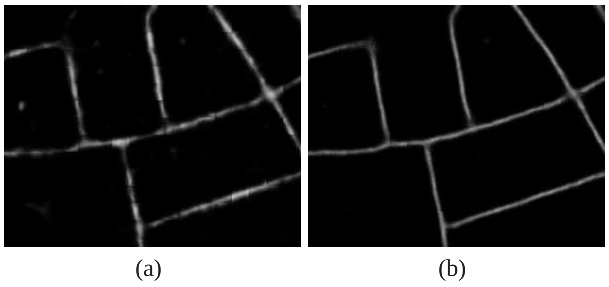


Fig. 3. (a) Predictions before post-processing. (b) Predictions after post-processing.

The neural network f_p is trained using stochastic gradient descent to minimize cross entropy between the ground truth map patches and the predictions as given by Equation (5). We do not use pretraining when training f_p, as this did not improve performance. As with training of the neural network f, we randomly rotate each training case before it is processed in order to remove a bias towards roads in some orientations.

The post-processing procedure is similar to the approach employed by Jain and Seung [15] for natural image denoising. They train a convolutional neural network to predict small noise-free patches of natural images given larger patches that had noise added to them. Since our post-processing procedure repeatedly applies a local filter at fixed intervals over a larger image, it can be seen as a type of convolutional neural network where the convolution is followed by subsampling. Jain and Seung show that this kind of neural network architecture can be seen as performing approximate inference in a special kind of Markov Random Field model [15]. Jain and Seung also show that this approach outperforms approaches based on Markov Random Fields on the image denoising task.

Figure 3(b) shows the result of applying the post-processing procedure to the predictions from figure 3(a). The process clearly removes disconnected blotches, fills in the gaps in the roads, and generally improves the quality of the predictions. While we do not do so in this paper, the post-processing procedure can be applied repeatedly, with each application receiving the predictions made by the previous application as input. This process propagates confident predictions along the predicted road network.

5 Experiments

We performed experiments on two datasets consisting of urban aerial imagery at a resolution of 1.2 meters per pixel. We will refer to the datasets as URBAN1 and URBAN2. Dataset URBAN1 covers a large metropolitan area with both urban and suburban regions. It consist of a training set that covers roughly 500 square kilometers, a separate test set of 50 square kilometers, and a separate small validation set that was used for model selection. Dataset URBAN2 is only used for testing and consists of 28 square kilometers of aerial imagery of a city different from the one covered in URBAN1. When generating the ground truth pixel labels as described in Section 2.1, the smoothing parameters σ was set to 2 pixels. This makes the area within one standard deviation of a pixel roughly 20 feet in diameter, which is approximately the width of a typical two lane road.

We made predictions for 16×16 map patches from 64×64 colour RGB aerial image patches, which corresponds to $w_m = 16$ and $w_s = 64$. The neural network f had 4096 input units, 12288 hidden units, and 256 output units. For the post-processing procedure, we set w_c to 64 and used 4096 hidden units in the neural net f_p. Hence f_p had 4096 input units, 4096 hidden units, and 256 output units[3]. All inputs to the neural networks were shifted and rescaled to have mean 0 and standard deviation 1.

Although our method is not overly sensitive to the parameter values, we present them here for completeness. We used stochastic gradient descent with minibatches of size 64 and momentum of 0.9 for training the neural networks. We used a learning rate of 0.0005 and L_2 weight decay of 0.0002. When training Restricted Boltzmann Machines we used the contrastive divergence approximation to the gradient [13]. Once again, we used stochastic gradient descent with minibatches of size 64 and momentum of 0.9. We used a learning rate of 0.001 and L_2 weight decay of 0.0002. We made between 10 and 20 passes through the training set when training the neural networks and RBMs.

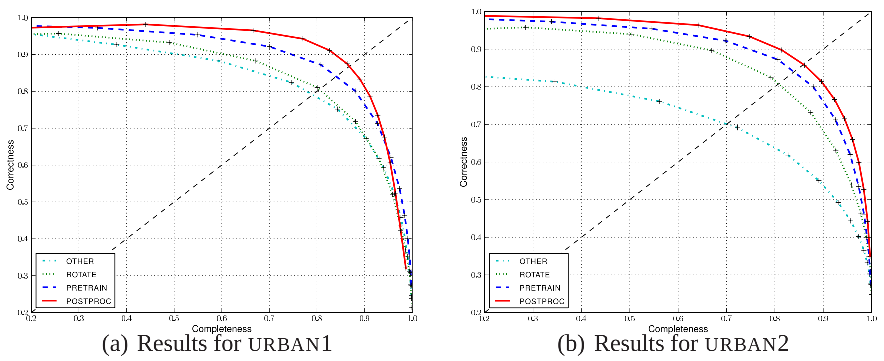
Since the models we have just described all have millions of parameters and the training set for dataset URBAN1 consists of over 1.2 million training cases, training our models would normally take months on a single core CPU or weeks on a multicore machine. We were able to train our best model in less than 3 days on a consumer GPU. This included pretraining and training of neural network f and training of the post-processing neural network f_p. Since the training procedures for neural networks and RBMs are easily expressed in terms of elementary matrix operations, porting them to the GPU was trivial. In both cases, we obtained speedups of more than an order of magnitude over the same algorithms running on a modern four-core CPU[4]. In order to implement the required algorithms on the GPU, we first created a GPU-based matrix library for Python. The CUDAMat library as well as our implementations of neural networks and RBMs are now available as open-source software [16].

5.1 Metrics

The most common metrics for evaluating road detection systems are correctness and completeness [17]. The *completeness* of a set of predictions is the fraction of true roads that were correctly detected, while the *correctness* is the fraction of predicted roads that are true roads. Since the road centreline locations that we used to generate ground truth are often noisy we compute relaxed completeness and correctness scores. Namely, in our experiments completeness represents the fraction of true road pixels that are within ρ pixels of a predicted road pixel, while correctness measures the fraction of predicted road pixels that are within ρ pixels of a true road pixel. Relaxing the completeness and

[3] Multiples of 64 were used because using arrays with dimensions that are multiples of 64 can help reduce the number of idle cores on the GPU.

[4] CPU implementations used parallel linear algebra routines and MATLAB.



(a) Results for URBAN1 (b) Results for URBAN2

Fig. 4. Completeness/correctness curves on URBAN1 and URBAN2.

correctness measures in this manner is common practice when evaluating road detection systems [17]. In this paper we set ρ to 3 pixels.

5.2 Results

Since our models provide us with road/non-road probabilities for map pixels, we need to select a threshold to make concrete predictions. For this reason we evaluate our models using completeness/correctness curves. Figure 4 shows completeness/correctness curves for the four models we evaluated on both datasets.

To compare to previous approaches, we evaluate a model, labelled OTHER, that uses a smaller context of size 24 and does not use rotated training data, pretraining, or post-processing. This approach has been used in several road detection systems [6, 7, 9], but with far less training data. The model OTHER is also an example of the kind of road detection system that can be trained on a modern CPU in the time it takes us to train our best model on a GPU.

We compare OTHER to three new models that used a context size of 64 and were trained as described above. The model ROTATE did not utilize pretraining or post-processing and is meant to show the performance of using a large context with rotated training data. The model PRETRAIN is a pretrained version of ROTATE. Finally, the model POSTPROC is the model PRETRAIN followed by our post-processing procedure.

The large difference in the performance of the model OTHER on the two datasets can be explained by the structure of their road networks. Many cities have large areas where the road network consists of a grid at some orientation, resulting in roads having two dominant orientations. Indeed, large parts of the cities in URBAN1 and URBAN2 consist of grids, however, the orientation of the grids is different between the two datasets. Since the model OTHER is trained on patches of URBAN1 without randomly rotating them, the model strongly favors roads in orientations similar to those in URBAN1. Since the dominant orientations of roads in URBAN2 are different, the performance of OTHER on URBAN2 is much worse than on URBAN1. This gap in performance shows that any approach that learns to detect roads from patches without incorporating rotations into the data or rotation invariance into the model is likely to work very poorly unless it is trained and tested on very similar conditions. This effect also highlights the importance of evaluating road detection systems on large datasets with a wide variety of road types and orientations.

Since the remaining three models randomly rotate each training case before processing it, our models exhibit similar performance on URBAN1 and URBAN2, suggesting that they are robust to significant variations between training and testing data. The results also show that unsupervised pretraining significantly improves road detection performance. If we compare the models by their break-even points, i.e. the points on the curves where completeness equals correctness, then unsupervised pretraining improves both completeness and correctness by about 0.05 on both datasets. The post-processing procedure further improves completeness and correctness on both datasets by approximately another 0.02.

Figure 5 presents a qualitative comparison between the typical predictions of the models OTHER and POSTPROC on the URBAN1 test set. Figure 5(a) shows that while OTHER is able to detect two-lane suburban roads quite well, the model often has problems with bigger roads. Figure 5(b) shows that the model POSTPROC is able to deal with wider roads. Figures 5(c) and 5(d) show the predictions of OTHER and POSTPROC respectively for an area that includes a highway interchange. The model OTHER clearly has trouble detecting the highway while POSTPROC does not.

To get a better understanding of the kinds mistakes our best model makes, POSTPROC consider Figure 6. It shows predictions made by the POSTPROC model on two regions taken from the URBAN1 test set. Figure 6(a) shows some typical examples of false positive detections. Most of the false positives are in fact paved regions that cars drive on. Since only named streets tend to be included in road maps, things like alleys and parking lots are not included and hence end up being labelled as false positives, if detected.

Figure 6(b) shows some examples of typical false negative detections, which tend to be caused by rare road types or conditions. For example, while our model is able to deal with shadows and occlusions caused by small objects, such as trees, it is unable to deal with shadows and occlusions caused by large buildings. One possible way of dealing with such problems is modifying the post-processing procedure to receive predictions as well as a satellite image patch of the same area as input. This should allow the post-processor to learn to fill in such gaps based on appearance.

We stress that our evaluation was performed on challenging urban data and covered an area roughly an order of magnitude larger than the areas used to evaluate previous work on road detection. We believe that our approach is the first to be shown to work reliably on real-world data on a large scale.

6 Related Work

Most of the prior work on road detection, starting with the initial work of Bajcsy and Tavakoli [1], follows an ad-hoc approach. A popular approach involves first extracting edges or other primitives and then applying grouping and pruning techniques to obtain the final road network. Laptev et al. [5] use scale space theory to extract a coarse road

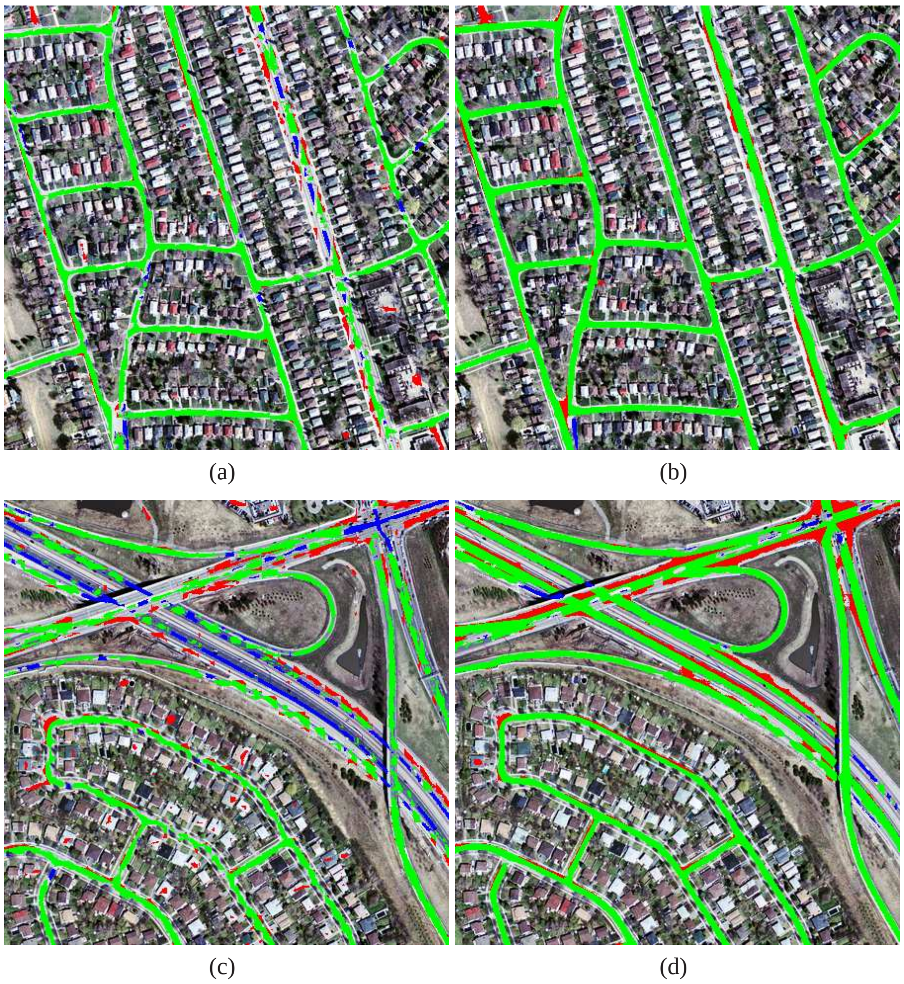


Fig. 5. a) and c) Visualization of the predictions made by OTHER. b) and d) Visualizations of the predictions made by POSTPROC. See the electronic version for colour. True positives are shown in green, false positives are shown in red, false negatives are shown in blue, and the background colour is used for true negatives. We used the threshold that corresponds to the break-even point on the completeness/correctness curves.

network and then apply a ribbon snake model to refine the road network, while Mena and Malpica [18] use segmentation followed by skeleton extraction. Another common strategy involves tracking roads from either expert-provided or automatically extracted starting points [19, 4].

One of the earliest attempts to learn to detect roads in aerial imagery is due to Boggess [7]. A neural network was used to predict road/non-road labels for a pixel given a small (5×5 pixels) aerial image context. Not surprisingly such a small context

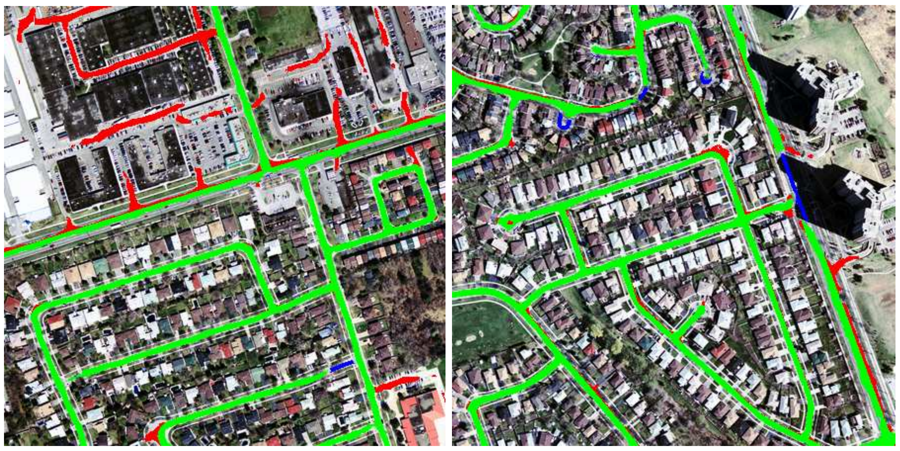


Fig. 6. Failure modes of the model POSTPROC. See the electronic version for colour.

is not sufficient for detecting roads in a wide variety of settings. Subsequent attempts to use neural networks for road detection [6, 9] did not achieve significant improvements over the results of Boggess as they also relied on a small context (9×9 pixels being the largest) for prediction and used very little training data.

Dollar et al. [10] presented some results on road detection for their general approach to learning object boundaries. They extract tens of thousands of predefined features (such as Haar filter responses) from a large context around each pixel and use a probabilistic boosting tree to make predictions. However, they only offer a proof-of-concept qualitative evaluation on three small images. While our approach shares many of the same characteristics, the key difference is that we learn the features and exploit the dependencies among the labels.

There is a vast literature on methods for exploiting dependencies among pixel labels to which our post-processing procedure is related. He et al. [20] applied Conditional Random Fields (CRFs) to the image labelling problem after extending them to the image domain. In the road detection literature, active contour models are often used to incorporate prior knowledge about the structure of road networks for improved detection results [5, 21]. Porway et al. [22] used a grammar to model relationships between objects such as cars, trees, and roofs for the purpose of parsing aerial images. As we have already mentioned, our post-processing step is similar to the approach of Jain and Seung [15] to image denoising. One advantage of this type of approach over using MRFs and CRFs with unrestricted potentials is that it avoids the need for performing approximate inference by directly learning a mapping.

7 Future Directions

The Gaussian-binary RBM that was used to initialize the feature-detecting layer of the neural network is not a very good generative model of images because it assumes that the pixels are independent given the features. A better generative model would include an explicit representation of the covariance structure of the image. This has been shown to improve discriminative performance for an object recognition task [23].

Most of the "errors" in the current system are due to the ambiguous nature of the labelling task. Our system often finds real roads that are simply not large enough to be labelled as roads by an expert. The use of vector maps that lack road width information also means that our system is penalized for correctly finding road pixels in wide roads such as highways. In addition to hurting the test performance, errors of this type hurt the training because the network is trying to fit inconsistent labels. A better way to handle ambiguous labels during training is to view the labels extracted from the map as noisy versions of an underlying set of true labels. This allows the neural network to override labels that are clearly incorrect during training.

8 Conclusions

We have presented an approach for automatically detecting roads in aerial imagery using neural networks. By using synthetic road/non-road labels and a consumer GPU board we were able to efficiently train much larger neural networks on much more data than was feasible before. We also showed how unsupervised pretraining and supervised post-processing substantially improves the performance of our road detector. The resulting road detection system works reliably on two large datasets of challenging urban data. To the best of our knowledge, no other published road detection system has been shown to work well on challenging urban data on such a scale.

References

1. Bajcsy, R., Tavakoli, M.: Computer recognition of roads from satellite pictures. IEEE Transactions on Systems, Man, and Cybernetics **6** (1976) 623–637
2. Baltsavias, E.P.: Object extraction and revision by image analysis using existing geodata and knowledge: current status and steps towards operational systems. ISPRS Journal of Photogrammetry and Remote Sensing **58** (2004) 129–151
3. Mayer, H.: Object extraction in photogrammetric computer vision. ISPRS Journal of Photogrammetry and Remote Sensing **63** (2008) 213–222
4. Hu, J., Razdan, A., Femiani, J.C., Cui, M., Wonka, P.: Road Network Extraction and Intersection Detection From Aerial Images by Tracking Road Footprints. IEEE Transactions on Geoscience and Remote Sensing **45** (2007) 4144–4157
5. Laptev, I., Mayer, H., Lindeberg, T., Eckstein, W., Steger, C., Baumgartner, A.: Automatic extraction of roads from aerial images based on scale space and snakes. Machine Vision and Applications **12** (2000) 23–31
6. Bhattacharya, U., Parui, S.K.: An improved backpropagation neural network for detection of road-like features in satellite imagery. International Journal of Remote Sensing **18** (1997) 3379–3394

7. Boggess, J.E.: Identification of roads in satellite imagery using artificial neural networks: A contextual approach. Technical report, Mississippi State University (1993)
8. Huang, X., Zhang, L.: Road centreline extraction from high-resolution imagery based on multiscale structural features and support vector machines. International Journal of Remote Sensing **30** (2009) 1977–1987
9. Mokhtarzade, M., Zoej, M.J.V.: Road detection from high-resolution satellite images using artificial neural networks. International Journal of Applied Earth Observation and Geoinformation **9** (2007) 32–40
10. Dollar, P., Tu, Z., Belongie, S.: Supervised learning of edges and object boundaries. In: CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. (2006) 1964–1971
11. Hinton, G., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science **313** (2006) 504 – 507
12. Larochelle, H., Bengio, Y., Louradour, J., Lamblin, P.: Exploring strategies for training deep neural networks. Journal of Machine Learning Research **10** (2009) 1–40
13. Hinton, G.: Training products of experts by minimizing contrastive divergence. Neural Computation **14** (2002) 1771–1800
14. Norouzi, M., Ranjbar, M., Mori, G.: Stacks of convolutional restricted boltzmann machines for shift-invariant feature learning. In: CVPR. (2009)
15. Jain, V., Seung, S.: Natural image denoising with convolutional networks. In Koller, D., Schuurmans, D., Bengio, Y., Bottou, L., eds.: Advances in Neural Information Processing Systems 21. (2009) 769–776
16. Mnih, V.: Cudamat: a CUDA-based matrix class for python. Technical Report UTML TR 2009-004, Department of Computer Science, University of Toronto (2009)
17. Wiedemann, C., Heipke, C., Mayer, H., Jamet, O.: Empirical evaluation of automatically extracted road axes. In: Empirical Evaluation Techniques in Computer Vision. (1998) 172–187
18. Mena, J.B., Malpica, J.A.: An automatic method for road extraction in rural and semi-urban areas starting from high resolution satellite imagery. Pattern Recognition Letters **26** (2005) 1201–1220
19. Geman, D., Geman, D., Jedynak, B., Jedynak, B., Syntim, P.: An active testing model for tracking roads in satellite images. IEEE Transactions on Pattern Analysis and Machine Intelligence **18** (1995) 1–14
20. He, X., Zemel, R.S., Carreira-Perpiñán, M.Á.: Multiscale conditional random fields for image labeling. In: CVPR '04: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. (2004) 695–702
21. Peng, T., Jermyn, I., Prinet, V., Zerubia, J.: An extended phase field higher-order active contour model for networks and its application to road network extraction from vhr satellite images. In: ECCV08. (2008) 509–520
22. Porway, J., Wang, K., Yao, B., Zhu, S.C.: A hierarchical and contextual model for aerial image understanding. In: Computer Vision and Pattern Recognition, IEEE Computer Society Conference on. (2008)
23. Ranzato, M., Krizhevsky, A., Hinton, G.E.: Factored 3-way restricted boltzmann machines for modeling natural images. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. (2010)