LARGE-SCALE SUPERVISED LEARNING FOR 3D POINT CLOUD LABELING: SEMANTIC3D.NET

Timo Hackel[a], Jan D. Wegner[a], Nikolay Savinov[b], Lubor Ladicky[b], Konrad Schindler[a], Marc Pollefeys[b]

[a] IGP, ETH Zurich, Switzerland - (timo.hackel, jan.wegner, konrad.schindler)@geod.baug.ethz.ch
[b] CVG, ETH Zurich, Switzerland - (nikolay.savinov, lubor.ladicky, marc.pollefeys)@inf.ethz.ch

ABSTRACT:

In this paper we review current state-of-the-art in 3D point cloud classification, present a new 3D point cloud classification benchmark data set of single scans with over four billion manually labelled points, and discuss first available results on the benchmark. Much of the stunning recent progress in 2D image interpretation can be attributed to the availability of large amounts of training data, which have enabled the (supervised) learning of deep neural networks. With the data set presented in this paper, we aim to boost the performance of CNNs also for 3D point cloud labelling. Our hope is that this will lead to a breakthrough of deep learning also for 3D (geo-)data. The *semantic3D.net* data set consists of dense point clouds acquired with static terrestrial laser scanners. It contains 8 semantic classes and covers a wide range of urban outdoor scenes, including churches, streets, railroad tracks, squares, villages, soccer fields and castles. We describe our labelling interface and show that, compared to those already available to the research community, our data set provides denser and more complete point clouds, with a much higher overall number of labelled points. We further provide descriptions of baseline methods and of the first independent submissions, which are indeed based on CNNs, and already show remarkable improvements over prior art. We hope that *semantic3D.net* will pave the way for deep learning in 3D point cloud analysis, and for 3D representation learning in general.

1. INTRODUCTION

Neural networks have made a spectacular comeback in image analysis since the seminal paper of (Krizhevsky et al., 2012), which revives earlier work of (Fukushima, 1980, LeCun et al., 1989). Especially deep convolutional neural networks (CNNs) have quickly become the core technique for a whole range of learning-based image analysis tasks. The large majority of state-of-the-art methods in computer vision and machine learning now include CNNs as one of their essential components. Their success for image-interpretation tasks is mainly due to (i) easily parallelisable network architectures that facilitate training from millions of images on a single GPU and (ii) the availability of huge public benchmark data sets like *ImageNet* (Deng et al., 2009, Russakovsky et al., 2015) and *Pascal VOC* (Everingham et al., 2010) for rgb images, or *SUN RGB-D* (Song et al., 2015) for rgb-d data.

While CNNs have been a great success story for image interpretation, they have not yet made a comparable impact for 3D point cloud interpretation. What makes supervised learning hard for 3D point clouds is the sheer size of millions of points per data set, and the irregular, not grid-aligned, and in places very sparse distribution of the data, with strongly varying point density (Figure 1).

While recording point clouds is nowadays straight-forward, the main bottleneck is to generate enough manually labeled training data, needed for contemporary (deep) machine learning to learn good models, that generalize well across new, unseen scenes. Due to the additional dimension, the number of classifier parameters is larger in 3D space than in 2D, and specific 3D effects like occlusion or variations in point density lead to many different patterns for identical output classes. This makes it harder to train good classifiers, so it can be expected that even more training data that in 2D is needed[1]. In contrast to images, which are fairly easy to annotate even for untrained users, 3D point clouds are harder to interpret. Navigation in 3D is more time-consuming and the strongly varying point density aggravates scene interpretation.

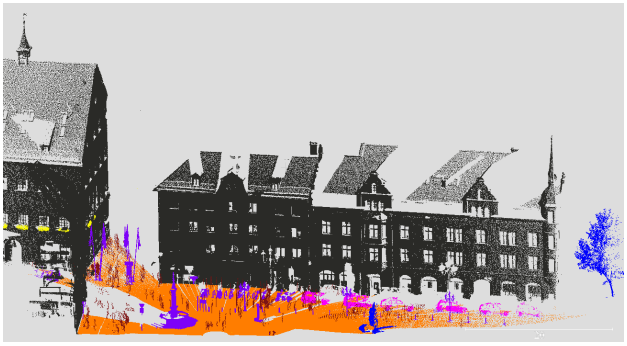


Figure 1: Example point cloud from the benchmark dataset, where colours indicate class labels.

In order to accelerate the development of powerful algorithms for point cloud processing[2], we provide the (to our knowledge) hitherto largest collection of individual, non-overlapping terrestrial laser scans with point-level semantic ground truth annotation. In total, it consists of over $4 \cdot 10^9$ points, labelled into 8 classes. The data set is split into training and test sets of approximately equal size, without any overlap between train and test scenes. The scans are challenging, not only due to their realistic size of up to $\approx 4 \cdot 10^8$ points per scan, but also because of their high angular resolution and long measurement range, leading to extreme density changes and large occlusions. For convenient use of the benchmark, we provide not only freely available data and ground truth, but also an automated online submission system, as well as

[1] The number of 3D points of *semantic3d.net* (4×10^9 points) is at the same scale as the number of pixels of the *SUN RGB-D* benchmark ($\approx 3.3 \times 10^9$ px) (Song et al., 2015), which aims at 3D object classification. However, the number of 3D points per laser scan ($\approx 4 \times 10^8$ points), and thus the variability in point density, object scale etc. is considerably larger than the number of pixels per image ($\approx 4 \times 10^5$ px).

[2] Note that, besides laser scanner point clouds, it is also sometimes preferred to classify point clouds generated via structure-from-motion directly instead of going back to the individual images and then merging the results (Riemenschneider et al., 2014).

evaluation tables for the submitted methods. The benchmark also includes baselines, both for the conventional pipeline consisting of eigenvalue-based feature extraction at multiple scales followed by classification with a random forest, and for a basic deep learning approach. Moreover, we briefly discuss the first submissions to the benchmark, which so far all employ deep learning. This article is an extended version of the conference paper (Hackel et al., 2017). Here, we add a more thorough review of related work, with emphasis on the most recent 3D-CNN methods. We also provide descriptions of the two latest CNN-based submissions, which lead the comparison by a significant margin, and seem to confirm that, also for point cloud anaysis, deep learning is the most powerful technology developed to date.

2. RELATED WORK

Here, we first review traditional methods for point cloud segmentation before discussing novel deep learning-based methods for this task. Finally, we review existing benchmark activities and motivate the introduction of our new 3D point cloud benchmark for semantic segmentation.

2.1 Point cloud segmentation

Early work on semantic point cloud segmentation transformed the points (recorded from airborne platforms) into other representations such as regular raster height maps, in order to simplify the problem and benefit from the comprehensive toolbox of image processing functions (Hug and Wehr, 1997, Maas, 1999, Haala et al., 1998, Rottensteiner and Briese, 2002, Lodha et al., 2006). Much of the pioneering work on true 3D (i.e., not 2.5D) point cloud processing was developed to guide autonomous outdoor robots (Vandapel et al., 2004, Manduchi et al., 2005, Montemerlo and Thrun, 2006, Lalonde et al., 2006, Munoz et al., 2009b) that rely on laser scanners to acquire data of their surroundings.

In general, it is advantageous if scene interpretation directly operates on 3D points, both for aerial (Charaniya et al., 2004, Chehata et al., 2009, Niemeyer et al., 2011, Yao et al., 2011, Lafarge and Mallet, 2011, Lafarge and Mallet, 2012, Niemeyer et al., 2014, Yan et al., 2015) and for terrestrial data (Brodu and Lague, 2012, Weinmann et al., 2013, Dohan et al., 2015). Full 3D processing can handle data which cannot be reduced to height maps in a straight-forward manner, in particular terrestrial data generated from multiple scan positions, and mobile mapping data.

Training a good model requires an expressive feature set. A large number of 3D point descriptors has been developed, which typically encode geometric properties within the point's neighborhood, like surface normal orientation, surface curvature, e.t.c.. Popular descriptors are for example spin images (Johnson and Hebert, 1999), fast point feature histograms (FPFH) (Rusu et al., 2009) and signatures of histograms (SHOT) (Tombari et al., 2010). One drawback of these rich descriptors is their high computational cost. While computation time is not an issue for small point sets (e.g., sparse key points), it is a crucial bottleneck when *all* points in a large point cloud shall be classified. A faster alternative – again for range images rather than true 3D point clouds – is the NARF operator, which is popular for key point extraction and description in the robotics community (Steder et al., 2010, Steder et al., 2011). In order to achieve robustness against viewpoint changes, it explicitly models object contour information. A computationally cheaper alternative for full 3D point data are features derived from the 3D structure tensor of a point's neighbourhood (Demantké et al., 2011), and from the point distribution in oriented (usually vertical) cylinders (Monnier et al., 2012, Weinmann et al., 2013).

2.2 Deep learning for point cloud annotation

Neural networks (usually of the deep, convolutional network flavour) offer the possibility to completely avoid heuristic feature design and feature selection. They are at present immensely popular in 2D image interpretation. Recently, deep learning pipelines have been adapted to voxel grids (Lai et al., 2014, Wu et al., 2015, Maturana and Scherer, 2015) and RGB-D images (Song and Xiao, 2016), too. Being completely data-driven, these techniques have the ability to capture appearance (intensity) patterns as well as geometric object properties. Moreover, their multi-layered, hierarchical architecture has the ability to encode a large amount of contextual information. Deep learning in 3D has been proposed for a variety of applications in robotics, computer graphics, and computer vision. To the best of our knowledge, the earliest attempt that applies a 3D-CNNs on a voxel grid is (Prokhorov, 2010). The author classifies objects in LiDAR point clouds and improves classification accuracy despite limited amount of training data, by combining supervised and unsupervised training. More recent 3D-CNNs that operates on voxel grids include (Maturana and Scherer, 2015) for landing zone detection in 3D LiDAR point clouds, (Wu et al., 2015) for learning representations of 3D object shapes, and (Huang and You, 2016) to densely label LiDAR point clouds into 7 different object categories. A general drawback when directly applying 3D-CNNs to dense voxel grids derived from originally sparse point clouds is the huge memory overhead for encoding empty space. Computational complexity grows cubically with respect to voxel grid resolution, although high detail would only be needed at object surfaces.

Therefore, more recent 3D-CNNs exploit the sparsity commonly found in voxel grids. One strategy is to resort to an octree representation, where empty space (and potentially also large, geometrically simple object parts) are represented at coarser scales than object details (Riegler et al., 2017, Engelcke et al., 2017, Tatarchenko et al., 2017). Since the octree partitioning is a function of the object at hand, an important question is how to automatically adapt to new, previously unseen objects at test time. While (Riegler et al., 2017) assume the octree structure to be known at test time, (Tatarchenko et al., 2017) learn to predict the octree structure together with the labels. This allows generalization to unseen instances of a learned object category, without injecting additional prior knowledge.

Another strategy is to rely only on a small subset of the most discriminative points, while neglecting the large majority of less informative ones (Li et al., 2016, Qi et al., 2017a, Qi et al., 2017b). The idea is that the network learns how to select the most informative points from training data and aggregates information into global descriptors for object shapes via fully-connected layers. This allows for both shape classification and per-point labeling, while using only a small subset of points, resulting in significant speed and memory gains.

2.3 Benchmark initiatives for point clouds

Benchmarking efforts have a long tradition in the geospatial data community and particularly in ISPRS. Recent efforts include, for example, the *ISPRS-EuroSDR benchmark on High Density Aerial Image Matching*[3] that evaluates dense matching methods for oblique aerial images (Haala, 2013, Cavegn et al., 2014) and the *ISPRS Benchmark Test on Urban Object Detection and Reconstruction*, which contains several different challenges like semantic segmentation of aerial images and 3D object reconstruction (Rottensteiner et al., 2013, Rottensteiner et al., 2014).

[3] `http://www.ifp.uni-stuttgart.de/ISPRS-EuroSDR/ImageMatching/index.en.html`

In computer vision, very large benchmark datasets with millions of images have become standard for learning-based image interpretation. A variety of datasets have been introduced, many tailored for specific tasks, some serving as basis for annual challenges for several consecutive years (e.g., *ImageNet, Pascal VOC*). Datasets that aim at boosting research in image classification and object detection heavily rely on images downloaded from the internet. Web-based imagery has been a major driver of benchmarks because no expensive, dedicated photography campaigns have to be accomplished for dataset generation. This makes it possible to scale benchmarks from hundreds to millions of images, although often weakly annotated and with a considerable amount of label noise, that has to be taken into account when working with the data. Additionally, one can assume that internet images constitute a very general collection of images with less bias towards particular sensors, scenes, countries, objects etc.. This mitigates overfitting, and enables the training of rich, high-capacity models that nevertheless generalize well.

One of the first successful attempts to object detection in images at very large scale is *tinyimages*[4] with over 80 million small (32×32 px) images (Torralba et al., 2008). A milestone and still widely used dataset for semantic image segmentation is the famous Pascal VOC[5] dataset and challenge (Everingham et al., 2010), which has been used for training and testing many of the well-known, state-of-the-art algorithms today like (Long et al., 2015, Badrinarayanan et al., 2017). Another, more recent dataset is *MSCOCO*[6], which contains 300,000 images with annotations that allow for object segmentation, object recognition in context, and image captioning. One of the most popular benchmarks in computer vision today is *ImageNet*[7] (Deng et al., 2009, Russakovsky et al., 2015), which made Convolutional Neural Networks popular in computer vision (Krizhevsky et al., 2012). It contains $> 14 \times 10^6$ images organized according to the semantic WordNet hierarchy[8], where words are grouped into sets of cognitive synonyms.

The introduction of the popular, low-cost range sensor Microsoft Kinect gave rise to several large rgb-d image databases. Popular examples are the *NYU Depth Dataset V2*[9] (Silberman et al., 2012) and *SUN RGB-D*[10] (Song et al., 2015) that provide labeled rgb-d images for object segmentation and scene understanding. Compared to laser scanners, low-cost, structured-light rgb-d sensors have much shorter measurement range, lower resolution, and work poorly outdoors, due to interference of the sunlight with the projected infrared pattern.

To the best of our knowledge, no publicly available dataset with laser scans at the scale of the aforementioned vision benchmarks exists today. Thus, many recent Convolutional Neural Networks that are designed for Voxel Grids (Brock et al., 2016, Wu et al., 2015) resort to artificially generated data from the CAD models of ModelNet (Wu et al., 2015), a rather small, synthetic dataset. As a consequence, recent ensemble methods, e.g., (Brock et al., 2016), reach performance of over 97% on ModelNet10, which clearly indicates that the dataset is either too easy, or too small and already significantly overfitted.

Those few existing laser scan datasets are mostly acquired with mobile mapping devices or robots like *DUT1* (Zhuang et al., 2015a), *DUT2* (Zhuang et al., 2015b), or *KAIST* (Choe et al., 2013), which are small ($< 10^7$ points) and not publicly available. Public laser scan datasets include *Oakland* (Munoz et al., 2009a) ($< 2 \times 10^6$ points), the *Sydney Urban Objects* (De Deuge et al., 2013), *Paris-rue-Madame* (Serna et al., 2014) and data from the *IQmulus & TerraMobilita Contest* (Vallet et al., 2015). All have in common that they use 3D LIDAR data from mobile mapping vehicles, which provides a much lower point density than static scans, like ours. They are also relatively small and localised, and thus prone to overfitting. The majority of today's available point cloud datasets comes without a thorough, transparent evaluation that is publicly available on the internet, continuously updated and that lists all submissions to the benchmark.

With the *semantic3D.net* benchmark presented in this paper, we attempt to close this gap. It provides a much larger labelled 3D point cloud data set with approximately four billion hand-labeled points, comes with a sound evaluation, and continuously updates submissions. It is the first dataset that allows fully-fledged deep learning on real 3D laser scans, with high-quality, per-point supervision.

3. DATA

Our 30 published individual, non-overlapping terrestrial laser scans consist of in total $\approx$ 4 billion 3D points. Although we would have many more scans that overlap largely with the ones in our benchmark data set and would facilitate co-registration for large scenes, we prefer to keep this for a later extension. The main reason for publishing only individual scans is the huge size per scan (2.72 GB for the largest scan). For the same reason, we did not record multiple echoes per pulse. The data set is split into 15 scans for training that come with labels and 15 scans for testing, where labels are not publicly released and kept by the organizers (see parameters in Tab. 1 & 2). Submitted results on the test set are evaluated completely automatically on the server and repeated submissions are limited to discourage overfitting on the test set. Train and test data sets are always from different scenes to avoid biasing classifiers and ensure that we verify generalization capability. The data set contains urban and rural scenes, like farms, town halls, sport fields, a castle and market squares. We intentionally selected various different natural and man-made scenes to prevent overfitting of the classifiers. All of the published scenes were captured in Central Europe and depict urban or rural European architecture, as shown in Figure 2. Surveying-grade laser scanners were used for recording these scenes. Colorization was performed in a post processing step, by generating high-resolution cubemaps from co-rgistered camera images. In general, static laser scans have a very high resolution and are able to measure long distances with little noise. Especially compared to point clouds derived via structure-from-motion pipelines or Kinect-like structured light sensors, laser scanners deliver superior geometric data quality.

Scanner positions for data recording were selected as usually done in real field campaigns: only little scan overlap as needed for registration, so that scenes can be recorded in a minimum of time. This free choice of the scanning position implies that no prior assumption based on point density and on class distributions can be made. We publish up to 3 laser scans per scene that have small overlap. The relative position of laser scans at the same location was estimated from targets.

The choice of output classes in a benchmark, independent of downstream applications, is not obvious. Based on feedback from geo-spatial industry experts, we use the following 8 classes, which are considered useful for a variety of surveying applications: *(1)*

[4] http://groups.csail.mit.edu/vision/TinyImages/
[5] http://host.robots.ox.ac.uk/pascal/VOC/
[6] http://mscoco.org/
[7] http://www.image-net.org
[8] https://wordnet.princeton.edu/
[9] http://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html
[10] http://rgbd.cs.princeton.edu

Train data set	Number of points	Scene type	Description	Download size [GB]
bildstein1	$29'302'501$	rural	church in bildstein	0.20
bildstein3	$23'765'246$	rural	church in bildstein	0.17
bildstein5	$24'671'679$	rural	church in bildstein	0.18
domfountain1	$35'494'386$	urban	cathedral in feldkirch	0.28
domfountain2	$35'188'343$	urban	cathedral in feldkirch	0.25
domfountain3	$35'049'972$	urban	cathedral in feldkirch	0.23
untermaederbrunnen1	$16'658'648$	rural	fountain in balgach	0.17
untermaederbrunnen3	$19'767'991$	rural	fountain in balgach	0.17
neugasse	$50'109'087$	urban	neugasse in st. gallen	0.32
sg27_1	$161'044'280$	rural	railroad tracks	1.87
sg27_2	$248'351'425$	urban	town square	2.72
sg27_4	$280'994'028$	rural	village	1.59
sg27_5	$218'269'204$	suburban	crossing	1.25
sg27_9	$222'908'898$	urban	soccer field	1.22
sg28_4	$258'719'795$	urban	town square	1.40

Table 1: Parameters of the full resolution semantic-8 training data set. Identical names (left column) with different IDs identify scans of the same scene (but with very low overlap). All ground truth labels together have size 0.01 GB for download. All parameters are also provided on the benchmark website `http://www.semantic3d.net/view_dbase.php?chl=1`

Test data set	Number of points	Scene type	Description	Download size [GB]
stgallencathedral1	$28'181'979$	urban	cathedral in st. gallen	0.22
stgallencathedral3	$31'328'976$	urban	cathedral in st. gallen	0.22
stgallencathedral6	$32'342'450$	urban	cathedral in st. gallen	0.22
marketsquarefeldkirch1	$23'228'738$	urban	market square in feldkirch	0.17
marketsquarefeldkirch4	$22'760'334$	urban	market square in feldkirch	0.15
marketsquarefeldkirch7	$23'264'911$	urban	market square in feldkirch	0.15
birdfountain1	$36'627'054$	urban	fountain in feldkirch	0.25
castleblatten1	$152'248'025$	rural	castle in blatten	0.24
castleblatten5	$195'356'302$	rural	castle in blatten	0.70
sg27_3	$422'445'052$	suburban	houses	2.40
sg27_6	$226'790'878$	urban	city block	1.27
sg27_8	$429'615'314$	urban	city center	2.08
sg27_10	$285'579'196$	urban	town square	1.56
sg28_2	$170'158'281$	rural	farm	0.94
sg28_5	$269'007'810$	suburban	buildings	1.35

Table 2: Parameters of the full resolution semantic-8 testing data set. Identical names (left column) with different IDs identify scans of the same scene (but with very low overlap). All parameters are also provided on the benchmark website `http://www.semantic3d.net/view_dbase.php?chl=1`

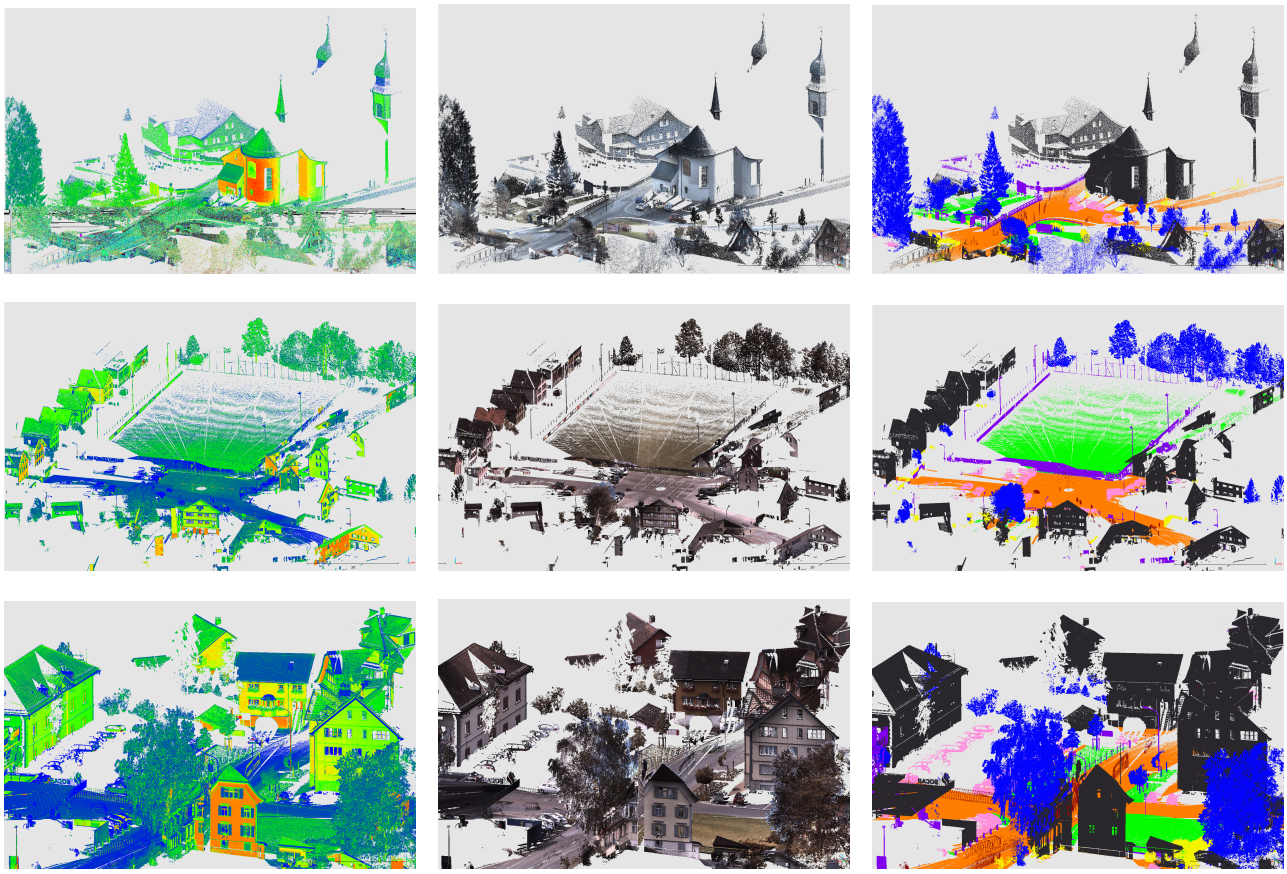


Figure 2: Intensity values (left), rgb colors (middle) and class labels (right) for example data sets.

man made terrain: mostly pavement; *(2) natural terrain:* mostly grass; *(3) high vegetation:* trees and large bushes; *(4) low vegetation:* flowers or small bushes which are smaller than 2 m; *(5) buildings:* Churches, city halls, stations, tenements, etc.; *(6) remaining hard scape:* a clutter class with for instance garden walls, fountains, benches, etc.; *(7) scanning artifacts:* artifacts caused by dynamically moving objects during the recording of the static scan; *(8) cars and trucks*. Some of these classes are ill-defined, for instance some scanning artifacts could also go for cars or trucks and it can be hard to differentiate between large and small bushes. Yet, we prefer not to alter the class nomenclature in a way that might reduce ambiguities, but departs from the requirements of the data providers and users. Note also, in many application projects class *7*, scanning artifacts, is filtered out in pre-processing with heuristic rule sets. Within the benchmark we prefer to also include that additional classification problem in the overall machine learning pipeline, and thus do not perform any heuristic pre-processing.

In our view, large data sets are important for two reasons: *a)* Typically, real world scan data are large. To have an impact on real problems, a method must be able to process large amounts of data. *b)* Large data sets are especially important for modern machine learning methods that involve representation learning (i.e., extracting discriminative low- to high-level features from the raw data). With too small data sets, good results leave strong doubts about possible overfitting; unsatisfactory results, on the other hand, are hard to interpret as guidelines for further research: are the mistakes due to short-comings of the method, or simply caused by insufficient training data?

3.1 Point Cloud Annotation

In contrast to common strategies for 3D data labelling that first compute an automatic over-segmentation and then label segments, we manually assign each point a class label individually. Although this strategy is more labor-intensive, it avoids inheriting errors from the segmentation; and, perhaps more importantly, it ensures that the ground truth does not contain any biases from a particular segmentation algorithm, that could be exploited by the classifier and impair its use with other training data. In general, it is more difficult for humans to label a point cloud by hand than images. The main problem is that it is hard to select a 3D point on a 2D monitor from a set of millions of points without a clear neighbourhood/surface structure. We tested two different strategies:

Annotation in 3D: We follow an iterative filtering strategy, where we manually select a couple of points, fit a simple model to the data, remove the model outliers and repeat these steps until all inliers belong to the same class. With this procedure it is possible to select large buildings in a couple of seconds. A small part of the point clouds was labeled with this approach by student assistants at ETH Zurich.

Annotation in 2D: The user rotates a point cloud, fixes a 2D view and draws a closed polygon which splits a point cloud into two parts (inside and outside of the polygon). One part usually contains points from the background and is discarded. This procedure is repeated a few times until all remaining points belong to the same class. In the end, all points are separated into different layers corresponding to classes of interest. This 2D procedure works well with existing software packages (Daniel Girardeau-Montaut, 2016) such that it can be outsourced to external labelers

more easily than the 3D work-flow. We used this procedure for all data sets where annotation was outsourced.

4. METHODS

Given a set of points (here: dense scans from a static, terrestrial laser scanner), we want to infer an individual class label per point. We provide three baseline methods that are meant to represent typical categories of approaches recently used for the task, covering the state of the art at the time of creating the benchmark.

4.1 2D Image Baseline

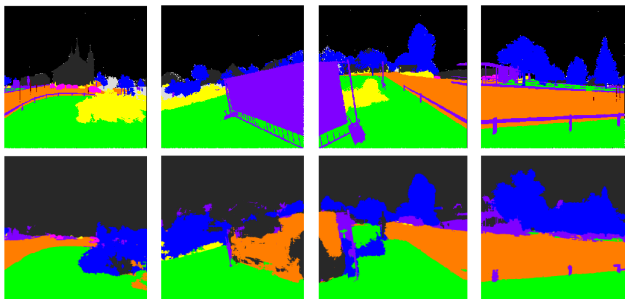


Figure 3: *Top row:* projection of ground truth to images. *Bottom row:* results of classification with the image baseline. *White:* unlabeled pixels, *black:* pixels with no corresponding 3D point, *gray:* buildings, *orange:* man made ground, *green:* natural ground, *yellow:* low vegetation, *blue:* high vegetation, *purple:* hard scape, *pink:* cars

We convert color values of the scans to separate images (without depth) with cube mapping (Greene, 1986). Cube maps are centered on the origin of the laser scanner and we thus do not experience any self-occlusions. Ground truth labels are also projected from the point clouds to image space, such that the 3D point labeling task turns into a purely image-based semantic segmentation problem in 2D (Figure 3). We chose the associative hierarchical random fields method (Ladicky et al., 2013) for semantic segmentation because it has proven to deliver good performance for a variety of tasks (e.g., (Montoya et al., 2014, Ladický et al., 2014)) and was available in its original implementation.

The method works as follows: four different types of features – textons (Malik et al., 2001), SIFT (Lowe, 2004), local quantized ternary patters (Hussain and Triggs, 2012) and self-similarity features (Shechtman and Irani, 2007) – are extracted densely at every image pixel. Each feature category is separately clustered into 512 distinct patterns using standard K-means clustering, which corresponds to a typical bag-of-words representation. For each pixel in an image, the feature vector is a concatenation of bag-of-word histograms over a fixed set of 200 rectangles of varying sizes. These rectangles are randomly placed in an extended neighbourhood around a pixel. We use multi-class boosting (Torralba et al., 2004) as classifier and the most discriminative weak features are found as explained in (Shotton et al., 2006). To add local smoothing without loosing sharp object boundaries, the model includes soft constraints that favor constant labels inside superpixels and class transitions at their boundaries. Super-pixels are extracted via mean-shift (Comaniciu and Meer, 2002) with 3 sets of coarse-to-fine parameters as described in (Ladicky et al., 2013). Class likelihoods of overlapping superpixels are predicted using the feature vector consisting of a bag-of-words representation for each superpixel. Pixel-based and superpixel-based classifiers with additional smoothness priors over pixels and superpixels are combined in a conditional random field framework, as proposed in (Kohli et al., 2008). The maximum a-posteriori label configuration is found using a graph-cut algorithm (Boykov and Kolmogorov, 2004), with appropriate graph construction for higher-order potentials (Ladicky et al., 2013).

4.2 3D Covariance Baseline

The second baseline was inspired by (Weinmann et al., 2015, Hackel et al., 2016). It infers the class label directly from the 3D point cloud using multiscale features and discriminative learning. Again, we had access to the original implementation of (Hackel et al., 2016). That method uses an efficient approximation of multi-scale neighbourhoods, where the point cloud is sub-sampled into a multi-resolution pyramid, such that a constant, small number of neighbours per level captures the multi-scale information. The multi-scale pyramid is generated by voxel-grid filtering with uniform spacing.

The feature set extracted at each level is an extension of the one decribed in (Weinmann et al., 2013). It uses different combinations of eigenvalues and eigenvectors of the covariance per point-neighborhood to represent geometric surface properties. Furthermore, height features based on vertical, cylindrical neighbourhoods are added to emphasize the special role of the gravity direction (assuming that scans are, as usual, aligned to the vertical). Note that we do not make use of color values or laser intensities. We empirically found that they did not improve the point cloud classification, moreover color or intensity information is not always available. As classifier, we use a random forest, for which optimal parameters (number of trees and tree depths) are found with grid search and five-fold cross-validation.

4.3 3D CNN Baseline

We design our baseline for the point cloud classification task following recent ideas of VoxNet (Maturana and Scherer, 2015) and ShapeNet (Wu et al., 2015) for 3D encoding. The pipeline is illustrated in Fig. 4. Instead of generating a global 3D voxel-grid prior to processing, we create $16 \times 16 \times 16$ voxel cubes per scan point[11]. We do this at 5 different resolutions, with voxel sizes

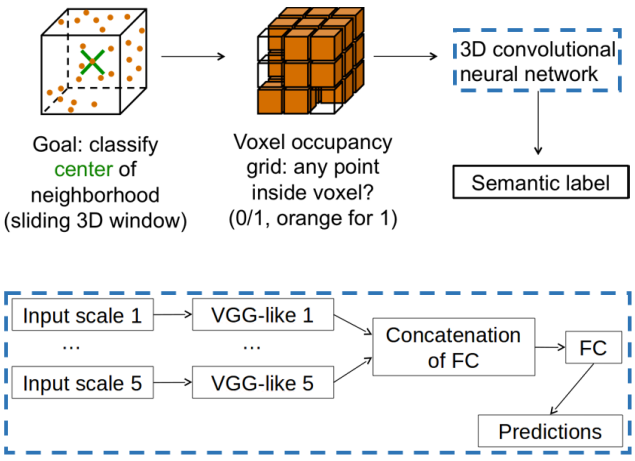


Figure 4: Our deep neural network baseline.

ranging from 2.5 cm to 40 cm (multiplied by powers of 2) and encode empty voxel cells as 0 and filled ones as 1. The input

[11] This strategy automatically centers each voxel-cube per scan point. Note that for the alternative approach of a global voxel grid, several scan points could fall into the same grid cell in dense regions of the scan. This would require scan point selection per grid cell, which is computationally costly and results in (undesired) down-sampling.

to the CNN is thus encoded in a multidimensional tensor with $5 \times 16 \times 16 \times 16$ cube entries per scan point.

Each of the five scales is handled separately by a VGG-like (Simonyan and Zisserman, 2015) network branch that includes convolutional, pooling and ReLU layers. The 5 separate network paths are finally concatenated into a single representation, which is passed through two fully-connected layers. The output of the second fully-connected layer is an 8-dimensional vector, which contains the class scores for each of the 8 classes in this benchmark challenge. Scores are transformed to class conditional probabilities with the soft-max function.

Before describing the network architecture in detail we introduce the following notation:

$c(i, o)$ stands for convolutional layers with $3 \times 3 \times 3$ filters, i input channels, o output channels, zero-padding of size 1 at each border and a stride of 1. $f(i, o)$ stands for fully-connected layers. r stands for a ReLU non-linearity, m stands for a volumetric max-pooling with receptive field $2 \times 2 \times 2$, applied with a stride of 2 in each dimension, d stands for a dropout with 0.5 probability, and s stands for a $softmax$ layer.

Our 3D CNN architecture assembles these components to a VGG-like network. We choose the filter size in convolutional layers as small as possible ($3 \times 3 \times 3$), as recommended in recent work (He et al., 2016), to have the least amount of parameters per layer and, hence, reduce both the risk of overfitting and the computational cost. Each of the 5 separate network paths, acting at different resolutions, has the sequence:

$$(c(1, 16), r, m, c(16, 32), r, m, c(32, 64), r, m).$$

The output is vectorized, concatenated across all branches (scales), and fed through two fully-connected layers to predict the class responses:

$$(f(2560, 2048), r, d, f(2048, 8), s).$$

The network is trained by minimising the standard multi-class cross-entropy loss, with stochastic gradient descent (SGD, (Bottou, 2010)). The SGD algorithm uses randomly sampled mini-batches of several hundred points per batch to iteratively update the parameters of the CNN. We use the popular *adadelta* (Zeiler, 2012) variant of SGD. We use a mini-batch size of 100 training samples (i.e., points), where each batch is sampled randomly and balanced to contain equal numbers of samples per class. We run training for 74,700 batches and sample training data from a large and representative point cloud with 259 million points (scan sg28_4). A standard pre-processing step for CNNs is data augmentation to enlarge the training set and to avoid overfitting. Here, we augment the training set with a random rotation around the z-axis after every 100 batches. During experiments it turned out that additional training data did not improve performance. This indicates that in our case we rather face underfitting (as opposed to overfitting), i.e., our model lacks the capacity to fully capture all the evidence in the available training data[12]. We thus refrain from further possible augmentations like randomly missing points or adding noise. The network is implemented in C++ and Lua and uses the Torch7 framework (Collobert et al., 2011) for deep learning. Code and documentation are available at `https://github.com/nsavinov/semantic3dnet`.

4.4 Submissions to the benchmark

The two top-performing approaches (Boulch et al., 2017, Lawin et al., 2017) submitted to the benchmark so far[13] both project 3D point clouds to 2D images, so as to harness the strength of well-established CNN models in 2D space. Their strategy is to: *(i)* render virtual 2D images from viewpoints in the 3D point cloud; *(ii)* perform semantic classification on the 2D images; *(iii)* lift the results back into 3D space, and merge the predictions from different 2D views. In the following, we provide a brief overview of both methods. Schematic work-flows are shown in Fig. 5 & 6.

The currently top-performing method is *SnapNet* (Boulch et al., 2017). The processing pipeline consists of four main parts (Fig. 5):

1) Point clouds are down-sampled with a voxel grid filter, 3D features are extracted (e.g., the deviation of surface normals to a vertical vector, sphericity etc.), and 3D meshes are generated by running the surface reconstruction approach of (Marton et al., 2009);

2) Virtual images are rendered from meshes at a high number (400 per point cloud for training) of different camera positions. RGB images as well as composite images with a channel for depth, the deviation of surface normals and sphericity are computed. For training and validation sets also virtual ground truth images are rendered. The authors propose to select camera view points either randomly in the bounding box of the scene (altitudes vary between 10 and 30 meters above ground) or to apply a multi-scale strategy, where three camera poses are generated for a subset of points that vary in distance to the selected point. A 3D mesh viewer renders virtual 2D images from the mesh.

3) Two different encoder-decoder CNNs, SegNet (Badrinarayanan et al., 2017) and U-Net (Ronneberger et al., 2015), are compared for semantic labeling of the rendered virtual images. Moreover, different strategies to combine RGB and depth information are tested, for example, model averaging and adding a shallow network to the output of the two separate depth and RGB networks.

4) Class responses of the neural network are back-projected to the mesh and averaged over the different virtual views. Finally, a kd-tree is used to assign the class label with the highest class response in the mesh to close points in the point cloud. The overall best results (i.e., those reported for the benchmark, cf. Tab. 3 & 4) are obtained with a combination of U-Net, shallow network for depth and RGB fusion, and multi-scale view generation.

The second-best submission at present is *DeePr3SS* (Lawin et al., 2017) (Fig. 6), which follows a coneptually similar strategy as (Boulch et al., 2017):

1) Virtual images with RGB channels as well as channels for depth and surface normals are rendered directly from the point clouds by point splatting (Zwicker et al., 2001) (which, unlike (Boulch et al., 2017), works without an intermediate mesh generation step). In total, 120 camera views are rendered per point cloud by rotating the camera around four vertical axis in the scene. Low quality images are discarded by using two filter strategies: First, images with a coverage below a threshold are removed. Second, views which are too close to large objects are neglected by thresholding the percentage of small depths.

2) Semantic segmentation is performed using fully convolutional networks, where the different inputs are fused by using a multi-stream architecture (Simonyan and Zisserman, 2014) that averages the output of the different streams. The authors use pre-trained VGG16 networks (Simonyan and Zisserman, 2015) for each stream and experiment with different combinations of streams for RGB, depth and normal channels. As often done, pre-training is performed on the ImageNet dataset (Russakovsky et al., 2015).

[12] Our model reaches the hardware limits of our GPU (TitanX with 12GB of RAM), we thus did not experiment with larger networks at this point.

[13] as of August 28, 2017

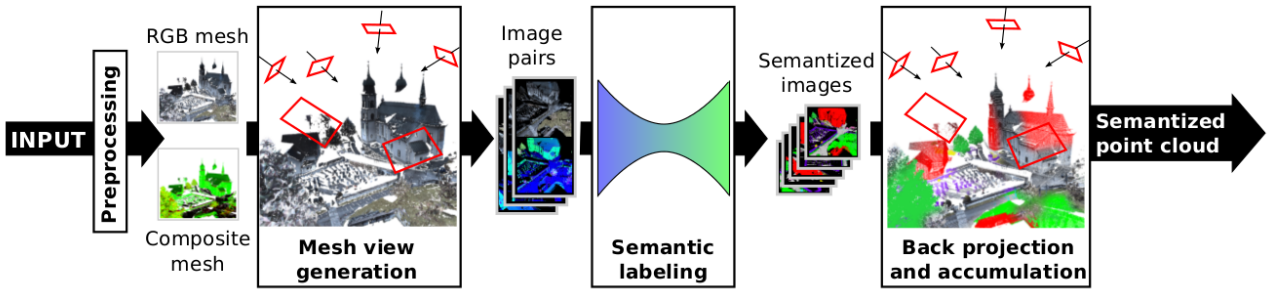


Figure 5: Work-flow of the *SnapNet* approach, figure taken from the original paper (Boulch et al., 2017).

3) Finally, class responses of the CNN are back-projected to the point cloud. Mapping between 3D points and pixels in the virtual images is given by rendering with point splatting. Class responses of the CNN for all pixels which correspond to the same 3D point are summed up, and the maximum average class response is used as final class label. The authors report that the multi-stream architecture with streams for all RGB, depth and normal channels works best (that workflow is used to produce numbers shown in Tab. 4 for the benchmark) for *DeePr3SS*.

5. EVALUATION

We follow the Pascal VOC challenge (Everingham et al., 2010) and choose the *Intersection over Union* (IoU), averaged over all classes, as our principal evaluation metric.[14]. Let the classes be indexed with integers from $\{1, \dots, N\}$, with N the number of different classes. Let C be an $N \times N$ confusion matrix of the chosen classification method, where each entry c_{ij} is a number of samples from ground-truth class i predicted as class j. Then the evaluation measure per class i is defined as

$$IoU_i = \frac{c_{ii}}{c_{ii} + \sum_{j \neq i} c_{ij} + \sum_{k \neq i} c_{ki}}. \quad (1)$$

The main evaluation measure of our benchmark is thus

$$\overline{IoU} = \frac{1}{N} \sum_{i=1}^{N} IoU_i. \quad (2)$$

We also report IoU_i for each class i and overall accuracy

$$OA = \frac{\sum_{i=1}^{N} c_{ii}}{\sum_{j=1}^{N} \sum_{k=1}^{N} c_{jk}} \quad (3)$$

as auxiliary measures and provide the confusion matrix C. Finally, each participant is asked to specify the time T it took to classify the test set as well as the hardware used for experiments. The computation time (if available) is important to understand how suitable the method is in real-world scenarios, where usually billions of points are required to be processed.

For computationally demanding methods we additionally provide a reduced challenge, consisting of a subset of the original test data. The results of our baseline methods as well as submissions are shown in Table 3 for the full challenge and in Table 4 for the reduced challenge. Of the three published baseline methods the classical machine learning pipeline with hand-designed, covariance-based features performs better than simplistic color image labeling without 3D information, and it also beats our simple CNN baseline, *DeepNet*. Due to its computational cost we could only run the *DeepNet* on the reduced data set. We note that *DeepNet* is meant as a baseline for "naive" application of CNNs to point cloud data, we do expect a more sophisticated, higher-capacity network to perform significantly better. Both *SnapNet* and *DeePr3SS* comfortably beat all baselines.

On the full challenge, two CNN methods, *SnapNet* and *HarrisNet* (unfortunately unpublished), already beat our best baseline by a significant margin (Table 3) of 12 respective 18 percent points. This indicates that deep learning seems to be the way to go also for point clouds, if enough training data is available. However, it should be noted that both *SnapNet* and *HarrisNet* are no true 3D-CNN approaches in the sense that they do not process 3D data directly. Both methods side-step 3D processing and cast semantic segmentation of point clouds as a 2D image labeling problem. For the future of the benchmark it will be interesting how true 3D-CNN approaches like (Riegler et al., 2017, Tatarchenko et al., 2017, Qi et al., 2017a) will perform. As a lesson learned, a future update of the benchmark should include multi-station point clouds that challenge the reprojection strategy.

6. BENCHMARK STATISTICS

Class distributions in the test and training sets are rather similar, as shown in Figure 7a. Interestingly, the class with most samples is *man-made terrain* because, out of convenience, operators in the field tend to place the scanner on flat and paved ground. Recall also the quadratic decrease of point density with distance to the scanner, such that many samples are close to the scanner. The largest difference between samples in test and training sets occurs for class *building*. However, this does not seem to affect the performance of the submissions so far. The most difficult classes, *scanning artefacts* and *cars*, have only few training and test samples and a large variation of possible object shapes. *Scanning artefacts* is probably the hardest class because the shape of artefacts mostly depends on the movement of objects during the scanning process. Note that, following discussions with industry professionals, the class *hard scape* was designed as a sort of "clutter class" that contains all sorts of man-made objects except for buildings, cars and the ground.

In order to quantify the quality of the manually acquired labels, we also checked the label agreement among human annotators. This provides an indicative measure how well different annotators agree on the correct labeling, and can be viewed as an internal check of manual labeling precision. To estimate the label agreement between different human annotators, we inspect areas where

[14] IoU compensates for different class frequencies as opposed to, for example, *overall accuracy* that does not balance different class frequencies, thus giving higher influence to large classes.

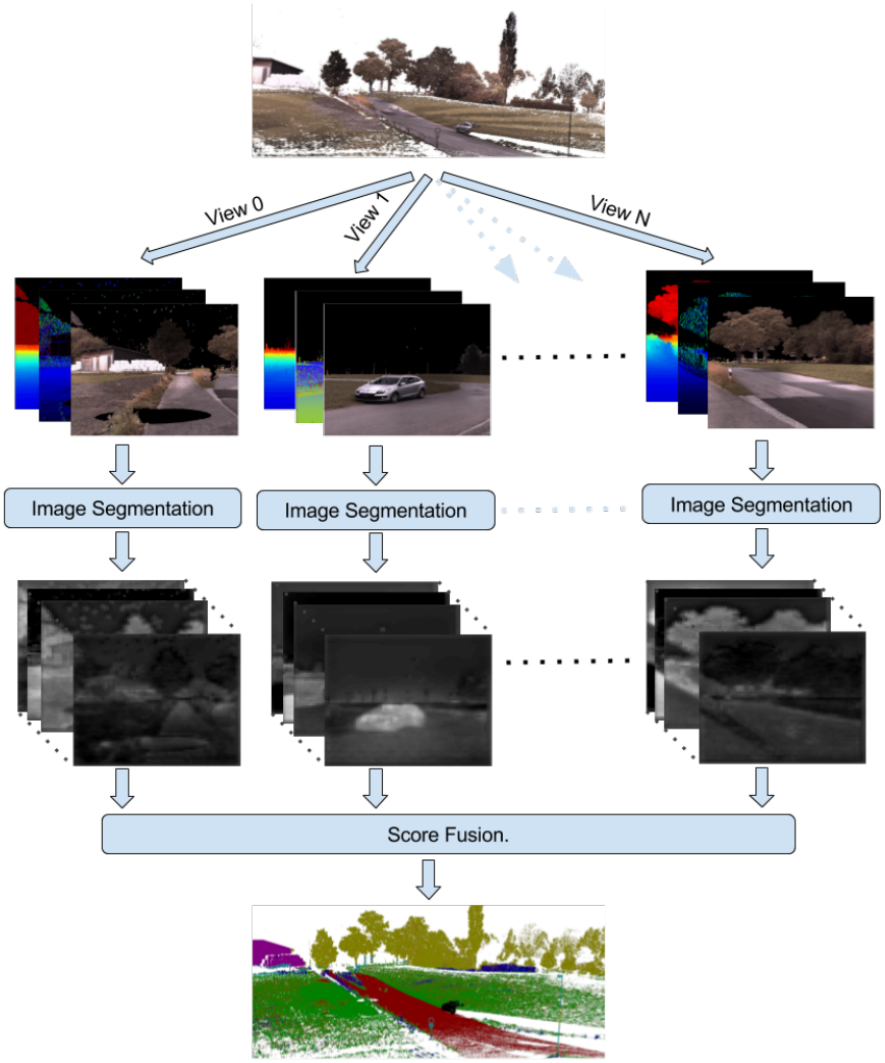
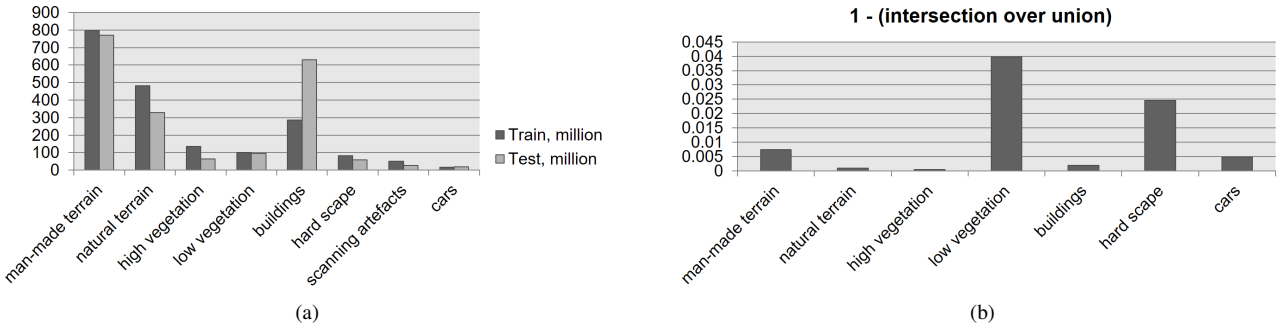


Figure 6: Work-flow of *DeePr3SS*, figure borrowed from the original paper (Lawin et al., 2017).



(a) (b)

Figure 7: (a) Number of points per class over all scans and (b) ground truth label errors estimated in overlapping parts of adjacent scans.

different scans of the same scene overlap (recall that overlaps of adjacent scans can be established precisely, via artificial markers placed in the scenes). Since we cannot rule out that some overlapping area might have been labeled twice by the same person (labeling was outsourced and we thus do not know exactly who annotated what), the observed consistency might in the worst case be slighty too optimistic. Even if scan alignments would be perfect without any error, no exact point-to-point correspondences exist between two scans, because scan points acquired from two different locations will not fall exactly onto the same 3D location. We thus have to resort to nearest-neighbor search to find point correspondences. Moreover, not all scan points have a corresponding point in the adjacent scan. A threshold of 5 cm on the distance is used to ignore those points where no correspondence exists. Once point correspondences have been established, it is possible to transfer the annotated labels from one point cloud to the other and compute a confusion matrix. Note that this definition of correspondence is not symmetric, "forward" point corre-

Method	$\overline{IoU}$	OA	$t[s]$	IoU_1	IoU_2	IoU_3	IoU_4	IoU_5	IoU_6	IoU_7	IoU_8
SnapNet	**0.674**	**0.910**	unknown	0.896	**0.795**	**0.748**	0.561	0.909	**0.365**	**0.343**	**0.772**
HarrisNet	0.623	0.881	unknown	0.818	0.737	0.742	**0.625**	**0.927**	0.283	0.178	0.671
TMLC-MS	0.494	0.850	**38421**	**0.911**	0.695	0.328	0.216	0.876	0.259	0.113	0.553
TML-PC	0.391	0.745	unknown	0.804	0.661	0.423	0.412	0.647	0.124	0.0*	0.058

Table 3: Semantic3d benchmark results on the full data set: 3D covariance baseline *TMLC-MS*, 2D RGB image baseline *TML-PC*, and first submissions *HarrisNet* and *SnapNet*. *IoU* for categories (1) man-made terrain, (2) natural terrain, (3) high vegetation, (4) low vegetation, (5) buildings, (6) hard scape, (7) scanning artefacts, (8) cars. * Scanning artefacts were ignored for 2D classification because they are not present in the image data.

Method	$\overline{IoU}$	OA	$t[s]$	IoU_1	IoU_2	IoU_3	IoU_4	IoU_5	IoU_6	IoU_7	IoU_8
SnapNet	**0.591**	0.886	3600	0.820	0.773	**0.797**	0.229	**0.911**	0.184	**0.373**	**0.644**
DeePr3SS	0.585	**0.889**	unknown	0.856	**0.832**	0.742	**0.324**	0.897	0.185	0.251	0.592
TMLC-MSR	0.542	0.862	**1800**	**0.898**	0.745	0.537	0.268	0.888	**0.189**	0.364	0.447
DeepNet	0.437	0.772	64800	0.838	0.385	0.548	0.085	0.841	0.151	0.223	0.423
TML-PCR	0.384	0.740	unknown	0.726	0.73	0.485	0.224	0.707	0.050	0.0*	0.15

Table 4: Semantic3d benchmark results on the reduced data set: 3D covariance baseline *TMLC-MSR*, 2D RGB image baseline *TML-PCR*, and our 3D CNN baseline *DeepNet*. *TMLC-MSR* is the same method as *TMLC-MS*, the same goes for *TMLC-PCR* and *TMLC-PC*. In both cases *R* indicates classifiers on the reduced dataset. *IoU* for categories (1) man-made terrain, (2) natural terrain, (3) high vegetation, (4) low vegetation, (5) buildings, (6) hard scape, (7) scanning artefacts, (8) cars. * Scanning artefacts were ignored for 2D classification because they are not present in the image data.

spondences from cloud A to cloud B are not in all cases the same as "backward" correspondences from cloud B to cloud A. For each pair, we calculate two intersection-over-union (IoU_i) values, which indicate negligible differences between forward and backward matching, an overall disagreement $< 3\%$, and a maximum label disagreement for the worst class (*low vegetation*) of $< 5\%$, see Figure 7b. Obviously, no correspondences between asynchronously acquired scans can be found on moving objects, so we ignored the class *scanning artefacts* in the evaluation.

7. CONCLUSION AND OUTLOOK

The *semantic3D.net* benchmark provides a large set of high quality, individual terrestrial laser scans with over 4 billion manually annotated points and a standardized evaluation framework. The data set has been published recently and the first results have been submitted. These already show that deep learning, and in particular appropriately adapted and well-engineered CNNs, outperform the leading conventional approaches, such as our covariance baseline, on large 3D laser scans. Interestingly, both top-performing methods SnapNet and HarrisNet are no true 3D-CNN approaches in the sense that they do not process 3D data directly. Both methods cast semantic point cloud segmentation as a 2D image labeling problem. This leaves room for methods that directly work in 3D and we hope to see more submissions of this kind in the future.

We are confident that, as more submissions appear, the benchmark will enable objective comparisons and yield new insights into strengths and weaknesses of different classification approaches for point clouds, and that the common testbed can help to guide future research efforts. We hope that the benchmark meets the needs of the research community and becomes a central resource for the development of new, more efficient and more accurate methods for semantic data interpretation in 3D space.

ACKNOWLEDGEMENT

This work is partially funded by the Swiss NSF project 163910, the Max Planck CLS Fellowship and the Swiss CTI project 17136.1 PFES-ES.

REFERENCES

Badrinarayanan, V., Kendall, A. and Cipolla, R., 2017. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 39(12), pp. 2481–2495.

Bottou, L., 2010. Large-scale machine learning with stochastic gradient descent. In: COMPSTAT'2010, Springer, pp. 177–186.

Boulch, A., Le Saux, B. and Audebert, N., 2017. Unstructured point cloud semantic labeling using deep segmentation networks. In: Eurographics Workshop on 3D Object Retrieval, The Eurographics Association, pp. 770–778.

Boykov, Y. and Kolmogorov, V., 2004. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(9), pp. 1124 –1137.

Brock, A., Lim, T., Ritchie, J. and Weston, N., 2016. Generative and discriminative voxel modeling with convolutional neural networks. In: 3D Deep Learning Workshop at NIPS 2016.

Brodu, N. and Lague, D., 2012. 3D terrestrial lidar data classification of complex natural scenes using a multi-scale dimensionality criterion: Applications in geomorphology. ISPRS Journal of Photogrammetry and Remote Sensing 68, pp. 121–134.

Cavegn, S., Haala, N., Nebiker, S., Rothermel, M. and Tutzauer, P., 2014. Benchmarking high density image matching for oblique airborne imagery. In: Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., Vol. XL-3, pp. 45–52.

Charaniya, A. P., Manduchi, R. and Lodha, S. K., 2004. Supervised parametric classification of aerial lidar data. In: IEEE Conference on Computer Vision and Pattern Recognition Workshop.

Chehata, N., Guo, L. and Mallet, C., 2009. Airborne lidar feature selection for urban classification using random forests. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., vol. 38, Part 3/W8, pp. 207–212.

Choe, Y., Shim, I. and Chung, M. J., 2013. Urban structure classification using the 3d normal distribution transform for practical robot applications. Advanced Robotics 27(5), pp. 351–371.

Collobert, R., Kavukcuoglu, K. and Farabet, C., 2011. Torch7: A matlab-like environment for machine learning. In: BigLearn, NIPS Workshop.

Comaniciu, D. and Meer, P., 2002. Mean shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(5), pp. 603–619.

Daniel Girardeau-Montaut, 2016. `http://www.danielgm.net/cc/`.

De Deuge, M., Quadros, A., Hung, C. and Douillard, B., 2013. Unsupervised feature learning for classification of outdoor 3d scans. In: Australasian Conference on Robitics and Automation, Vol. 2.

Demantké, J., Mallet, C., David, N. and Vallet, B., 2011. Dimensionality based scale selection in 3d lidar point clouds. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., vol. 38, Part 5/W12 pp. 97–102.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255.

Dohan, D., Matejek, B. and Funkhouser, T., 2015. Learning hierarchical semantic segmentations of lidar data. In: International Conference on 3D Vision, pp. 273–281.

Engelcke, M., Rao, D., Wang, D. Z., Tong, C. H. and Posner, I., 2017. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In: IEEE International Conference on Robotics and Automation, pp. 1355–1361.

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

Fukushima, K., 1980. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological cybernetics 36(4), pp. 193–202.

Greene, N., 1986. Environment mapping and other applications of world projections. IEEE Computer Graphics and Applications 6(11), pp. 21–29.

Haala, N., 2013. The landscape of dense image matching algorithms. In: Photogrammetric Week 13, pp. 271–284.

Haala, N., Brenner, C. and Anders, K.-H., 1998. 3d urban gis from laser altimeter and 2d map data. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci. 32, pp. 339–346.

Hackel, T., Savinov, N., Ladicky, L., Wegner, J. D., Schindler, K. and Pollefeys, M., 2017. SEMANTIC3D.NET: A new large-scale point cloud classification benchmark. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. IV-1-W1, pp. 91–98.

Hackel, T., Wegner, J. D. and Schindler, K., 2016. Fast semantic segmentation of 3D point clouds with strongly varying point density. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. III-3, pp. 177–184.

He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.

Huang, J. and You, S., 2016. Point Cloud Labeling using 3D Convolutional Neural Network. In: International Conference on Pattern Recognition, pp. 2670–2675.

Hug, C. and Wehr, A., 1997. Detecting and identifying topographic objects in imaging laser altimeter data. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci. 32(3/4W2), pp. 19–26.

Hussain, S. and Triggs, B., 2012. Visual recognition using local quantized patterns. In: European Conference on Computer Vision, pp. 716–729.

Johnson, A. E. and Hebert, M., 1999. Using spin images for efficient object recognition in cluttered 3d scenes. IEEE Transactions on pattern analysis and machine intelligence 21(5), pp. 433–449.

Kohli, P., Ladicky, L. and Torr, P. H. S., 2008. Robust higher order potentials for enforcing label consistency. In: IEEE Conference on Computer Vision and Pattern Recognition.

Krizhevsky, A., Sutskever, I. and Hinton, G. E., 2012. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems.

Ladicky, L., Russell, C., Kohli, P. and Torr, P., 2013. Associative hierarchical random fields. IEEE Transactions on Pattern Analysis and Machine Intelligence 36(6), pp. 1056–1077.

Ladický, L., Zeisl, B. and Pollefeys, M., 2014. Discriminatively trained dense surface normal estimation. In: European Conference on Computer Vision, pp. 468–484.

Lafarge, F. and Mallet, C., 2011. Building large urban environments from unstructured point data. In: IEEE International Conference on Computer Vision, pp. 1068–1075.

Lafarge, F. and Mallet, C., 2012. Creating large-scale city models from 3D-point clouds: a robust approach with hybrid representation. International Journal of Computer Vision 99(1), pp. 69–85.

Lai, K., Bo, L. and Fox, D., 2014. Unsupervised feature learning for 3d scene labeling. In: IEEE International Conference on Robotics and Automation, pp. 3050–3057.

Lalonde, J.-F., Vandapel, N., Huber, D. and Hebert, M., 2006. Natural terrain classification using three-dimensional ladar data for ground robot mobility. Journal of Field Robotics 23(10), pp. 839–861.

Lawin, F. J., Danelljan, M., Tosteberg, P., Bhat, G., Khan, F. S. and Felsberg, M., 2017. Deep projective 3d semantic segmentation. In: International Conference on Computer Analysis of Images and Patterns, LNCS 10424, Part I, Springer, Heidelberg, pp. 95–107.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. and Jackel, L. D., 1989. Backpropagation applied to handwritten zip code recognition. Neural computation 1(4), pp. 541–551.

Li, Y., Pirk, S., Su, H., Qi, C. R. and Guibas, L. J., 2016. Fpnn: Field probing neural networks for 3d data. In: D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon and R. Garnett (eds), Advances in Neural Information Processing Systems, Vol. 29, pp. 307–315.

Lodha, S., Kreps, E., Helmbold, D. and Fitzpatrick, D., 2006. Aerial LiDAR Data Classification using Support Vector Machines (SVM). In: IEEE Third International Symposium on 3D Data Processing, Visualization, and Transmission.

Long, J., Shelhamer, E. and Darrell, T., 2015. Fully convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440.

Lowe, D. G., 2004. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), pp. 91–110.

Maas, H.-G., 1999. The potential of height texture measures for the segmentation of airborne laserscanner data. In: Fourth international airborne remote sensing conference and exhibition/21st Canadian symposium on remote sensing, Vol. 1, pp. 154–161.

Malik, J., Belongie, S., Leung, T. and Shi, J., 2001. Contour and texture analysis for image segmentation. International Journal of Computer Vision 43(1), pp. 7–27.

Manduchi, R., A, C., Talukder, A. and Matthies, L., 2005. Obstacle Detection and Terrain Classification for Autonomous Off-Road Navigation. Autonomous Robots 18, pp. 81–102.

Marton, Z. C., Rusu, R. B. and Beetz, M., 2009. On fast surface reconstruction methods for large and noisy point clouds. In: IEEE International Conference on Robotics and Automation, pp. 3218–3223.

Maturana, D. and Scherer, S., 2015. Voxnet: A 3d convolutional neural network for real-time object recognition. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 922–928.

Monnier, F., Vallet, B. and Soheilian, B., 2012. Trees detection from laser point clouds acquired in dense urban areas by a mobile mapping system. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences I-3, pp. 245–250.

Montemerlo, M. and Thrun, S., 2006. Large-Scale Robotic 3-D Mapping of Urban Structures. In: M. Ang and O. Khatib (eds), Experimental Robotics IX. Springer Tracts in Advanced Robotics, Vol. 21, Springer, Heidelberg, pp. 141–150.

Montoya, J., Wegner, J. D., Ladický, L. and Schindler, K., 2014. Mind the gap: modeling local and global context in (road) networks. In: German Conference on Pattern Recognition, LNCS 8753, Springer, Heidelberg, pp. 212–223.

Munoz, D., Bagnell, J. A., Vandapel, N. and Hebert, M., 2009a. Contextual classification with functional max-margin markov networks. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 975–982.

Munoz, D., Vandapel, N. and Hebert, M., 2009b. Onboard Contextual Classification of 3-D Point Clouds with Learned High-order Markov Random Fields. In: IEEE International Conference on Robotics and Automation, pp. 2009–2016.

Niemeyer, J., Rottensteiner, F. and Soergel, U., 2014. Contextual classification of lidar data and building object detection in urban areas. ISPRS Journal of Photogrammetry and Remote Sensing 87, pp. 152–165.

Niemeyer, J., Wegner, J. D., Mallet, C., Rottensteiner, F. and Soergel, U., 2011. Conditional random fields for urban scene classification with full waveform lidar data. In: Photogrammetric Image Analysis, LNCS 6952, Springer, Heidelberg, pp. 233–244.

Prokhorov, D., 2010. A Convolutional Learning System for Object Classification in 3-D Lidar Data. IEEE Transactions on Neural Networks 21(5), pp. 858–863.

Qi, C. R., Su, H., Mo, K. and Guibas, L. J., 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 77–85.

Qi, C. R., Yi, L., Su, H. and Guibas, L. J., 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems.

Riegler, G., Ulusoy, A. O. and Geiger, A., 2017. Octnet: Learning deep 3d representations at high resolutions. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 6620–6629.

Riemenschneider, H., Bódis-Szomorú, A., Weissenberg, J. and Van Gool, L., 2014. Learning where to classify in multi-view semantic segmentation. In: European Conference on Computer Vision, Springer, pp. 516–532.

Ronneberger, O., Fischer, P. and Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, pp. 234–241.

Rottensteiner, F. and Briese, C., 2002. A new method for building extraction in urban areas from high-resolution lidar data. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci. 34(3/A), pp. 295–301.

Rottensteiner, F., Sohn, G., Gerke, M. and Wegner, J. D., 2013. ISPRS Test Project on Urban Classification and 3D Building Reconstruction. Technical report, ISPRS Working Group III / 4 - 3D Scene Analysis.

Rottensteiner, F., Sohn, G., Gerke, M., Wegner, J., Breitkopf, U. and Jung, J., 2014. Results of the ISPRS Benchmark on Urban Object Detection and 3D Building Reconstruction. ISPRS Journal of Photogrammetry and Remote Sensing 93, pp. 256–271.

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

Rusu, R. B., Marton, Z. C., Blodow, N., Holzbach, A. and Beetz, M., 2009. Model-based and learned semantic object labeling in 3d point cloud maps of kitchen environments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3601–3608.

Serna, A., Marcotegui, B., Goulette, F. and Deschaud, J.-E., 2014. Paris-rue-madame database: a 3d mobile laser scanner dataset for benchmarking urban detection, segmentation and classification methods. In: 4th International Conference on Pattern Recognition, Applications and Methods.

Shechtman, E. and Irani, M., 2007. Matching local self-similarities across images and videos. In: IEEE Conference on Computer Vision and Pattern Recognition.

Shotton, J., Winn, J., Rother, C. and Criminisi, A., 2006. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In: European Conference on Computer Vision, LNCS 3951, Part I, Springer, Heidelberg, pp. 1–15.

Silberman, N., Hoiem, D., Kohli, P. and Fergus, R., 2012. Indoor segmentation and support inference from rgbd images. In: European Conference on Computer Vision, Springer, pp. 746–760.

Simonyan, K. and Zisserman, A., 2014. Two-stream convolutional networks for action recognition in videos. In: Advances in neural information processing systems, pp. 568–576.

Simonyan, K. and Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations.

Song, S. and Xiao, J., 2016. Deep sliding shapes for amodal 3d object detection in rgb-d images. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 808–816.

Song, S., Lichtenberg, S. P. and Xiao, J., 2015. Sun rgb-d: A rgb-d scene understanding benchmark suite. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 567–576.

Steder, B., Rusu, R. B., Konolige, K. and Burgard, W., 2010. Narf: 3d range image features for object recognition. In: Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Vol. 44.

Steder, B., Rusu, R. B., Konolige, K. and Burgard, W., 2011. Point feature extraction on 3D range scans taking into account object boundaries. In: IEEE Int. Conf. on Robotics & Automation, pp. 2601–2608.

Tatarchenko, M., Dosovitskiy, A. and Brox, T., 2017. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. arXiv preprint arXiv:1703.09438.

Tombari, F., Salti, S. and Di Stefano, L., 2010. Unique signatures of histograms for local surface description. In: European Conference on Computer Vision, Springer, pp. 356–369.

Torralba, A., Fergus, R. and Freeman, W. T., 2008. 80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(11), pp. 1958–1970.

Torralba, A., Murphy, K. and Freeman, W., 2004. Sharing features: efficient boosting procedures for multiclass object detection. In: IEEE Conference on Computer Vision and Pattern Recognition.

Vallet, B., Brédif, M., Serna, A., Marcotegui, B. and Paparoditis, N., 2015. Terramobilita/iqmulus urban point cloud analysis benchmark. Computers & Graphics 49, pp. 126–133.

Vandapel, N., Huber, D., Kapuria, A. and Hebert, M., 2004. Natural Terrain Classification using 3-D Ladar Data. In: IEEE International Conference on Robotics and Automation, pp. 5117–5122.

Weinmann, M., Jutzi, B. and Mallet, C., 2013. Feature relevance assessment for the semantic interpretation of 3d point cloud data. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences II-5(W2), pp. 313–318.

Weinmann, M., Urban, S., Hinz, S., Jutzi, B. and Mallet, C., 2015. Distinctive 2d and 3d features for automated large-scale scene analysis in urban areas. Computers & Graphics 49, pp. 47–57.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X. and Xiao, J., 2015. 3d shapenets: A deep representation for volumetric shapes. IEEE Conference on Computer Vision and Pattern Recognition pp. 1912–1920.

Yan, W., Shaker, A. and El-Asjmawy, N., 2015. Urban land cover classification using airborne LiDAR data: A review. Remote Sensing of Environment 158, pp. 295–310.

Yao, W., Hinz, S. and Stilla, U., 2011. Extraction and motion estimation of vehicles in single-pass airborne lidar data towards urban traffic analysis. ISPRS Journal of Photogrammetry and Remote Sensing 66(3), pp. 260–271.

Zeiler, M. D., 2012. Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701.

Zhuang, Y., He, G., Hu, H. and Wu, Z., 2015a. A novel outdoor scene-understanding framework for unmanned ground vehicles with 3d laser scanners. Transactions of the Institute of Measurement and Control 37(4), pp. 435–445.

Zhuang, Y., Liu, Y., He, G. and Wang, W., 2015b. Contextual classification of 3d laser points with conditional random fields in urban environments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3908–3913.

Zwicker, M., Pfister, H., Van Baar, J. and Gross, M., 2001. Surface splatting. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, ACM, pp. 371–378.