Data for Image Recognition Tasks: An Efficient Tool for Fine-Grained Annotations

Marco Filax, Tim Gonschorek and Frank Ortmeier
Chair of Software Engineering, Otto von Guericke University, Magdeburg, Germany

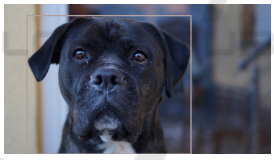
Keywords: Image and Video Analysis, Assistive Computer Vision, Fine-Grained Recognition, Dataset.

Abstract: Using large datasets is essential for machine learning. In practice, training a machine learning algorithm requires hundreds of samples. Multiple off-the-shelf datasets from the scientific domain exist to benchmark new approaches. However, when machine learning algorithms transit to industry, e.g., for a particular image classification problem, hundreds of specific purpose images are collected and annotated in laborious manual work. In this paper, we present a novel system to decrease the effort of annotating those large image sets. Therefore, we generate 2D bounding boxes from minimal 3D annotations using the known location and orientation of the camera. We annotate a particular object of interest in 3D once and project these annotations on to every frame of a video stream. The proposed approach is designed to work with off-the-shelf hardware. We demonstrate its applicability with an example from the real world. We generated a more extensive dataset than available in other works for a particular industrial use case: fine-grained recognition of items within grocery stores. Further, we make our dataset available to the interested vision community consisting of over 60,000 images. Some images were taken under ideal conditions for training while others were taken with the proposed approach in the wild.

1 INTRODUCTION

Deep learning is an emerging topic in science. Especially object recognition made essential use of data-driven approaches (Krizhevsky et al., 2012; Simonyan and Zisserman, 2015; Redmon et al., 2015). Through more extensive databases more powerful models can be trained (Deng et al., 2009). Recently, data-driven approaches began to move to industry. Frameworks like *TensorFlow*[1], *Caffe*[2], and *CNTK*[3] ease the hurdle of diving into the machine learning world. Besides a robust framework, lots of labeled data is required to apply a deep learning approach in a particular domain. Thereby, more labeled data typically implies better results. Especially for image recognition tasks, there already exist numerous general purpose datasets, e.g., ImageNet (Deng et al., 2009), SVHN (Netzer and Wang, 2011), Caltech-101 (Fei-Fei et al., 2006), or COCO (Lin et al., 2014). However, all of these datasets are typically applied to compare different academic recognition approaches and benchmark their performance.

[1]https://tensorflow.org
[2]https://caffe.berkeleyvision.org
[3]https://cntk.ai



(a) General Recognition: The dog covers the largest portion of the image.

(b) Fine-grained Recognition: Every item covers a small portion of the image.

Figure 1: The difference between recognition tasks in general and fine-grained recognition.

Migrating from academia to industry is more difficult since industrial applications typically are not covered by off-the-shelf datasets. A specialized data scientist must laboriously collect numerous images and manually annotate every image to learn a particular image classifier, which is time-consuming. In general recognition tasks, an image typically contains only a few elements that represent by a significant portion of the image. In fine-grained recognition tasks, dozens of elements cover small portions of the image. Figure 1 illustrates this problem. Data acquisition is especially a problem for fine-grained recognition tasks. It typically requires experts to distinguish the subtle differences between similar object classes. As a result, we

Filax, M., Gonschorek, T. and Ortmeier, F.
Data for Image Recognition Tasks: An Efficient Tool for Fine-Grained Annotations.
DOI: 10.5220/0007688709000907
In *Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2019)*, pages 900-907
ISBN: 978-989-758-351-3

can only rely on a few experts to acquire annotated learning data. The increased preparation time makes it problematic for industrial use cases.

It takes about one and a half minutes to label a single image for fine-grained tobacco recognition (Varol and Kuzu, 2014). This time scales with the number of elements in an image and objects in the database. That means that it would take more than three working days to annotate 1000 images with fine-grained bounding boxes in a similar product recognition. Building a groceries dataset with 60.000 annotated images with only a single expert is not reasonable. This observation summarizes the need for a system reducing the time required to annotate images.

In this work, we propose a system to annotate images semi-automatically. The core idea is to use a 3D simultaneous localization and mapping (SLAM) approach while sampling video streams. We minimize the effort with 3D labeling during acquisition and thereby decrease the effort to sample 2D bounding boxes. The proposed tool supports offline annotation refinements, e.g., for fine-grained recognition tasks. *DGen* is designed to work with off-the-shelf hardware: we use Microsoft's HoloLens. We demonstrate the applicability with a large, fine-grained grocery dataset. The second contribution is the *Magdeburg Groceries* dataset[4] itself. We describe the collected data and make our dataset publicly available.

The remainder of the paper is structured as follows: We describe our system and the proposed approach in section 2. Afterward, we describe the *Magdeburg Groceries* dataset in detail and evaluate the quality of the resulting dataset and the approach (cf. sect. 3). Further, we compare the proposed annotation procedure with the state-of-the-art. In section 4, we demonstrate the applicability of the proposed system by comparing our dataset to others from related works. Finally, we conclude our work.

2 THE DATASET GENERATOR

DGen is a tool to generate large datasets for visual recognition tasks. It aims at reducing the amount of time required to collect and annotate images for training and validating visual deep learning algorithms in an industrial setting. Methods like transfer learning (Pan and Yang, 2010) might ease the hunger for labeled data. Recently, a lot of weakly supervised approaches have been proposed, that aim at using labeled and unlabeled data (Zhou, 2017). However, these approaches typically require manually annotated samples as well. Thus, we found it necessary to propose a system to reduce the time needed to annotate images.

[4]https://bitbucket.org/cse_admin/md_groceries

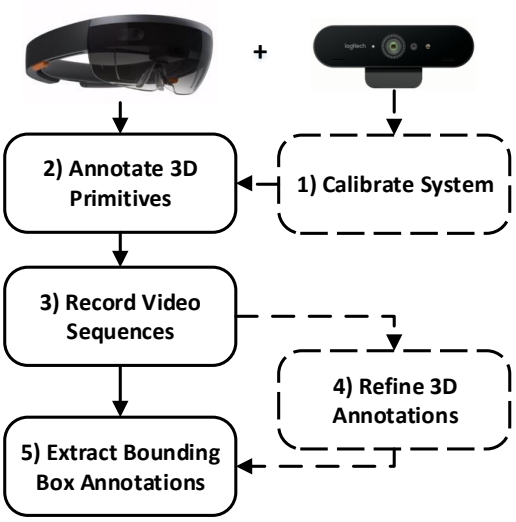


Figure 2: The *DGen* workflow at a glance. We use a SLAM approach to acquire 3D annotations and project their 2D position onto the recorded video sequence. We use Microsoft's HoloLens as an integrated solution for the SLAM and user inputs. Optionally, we support a dedicated camera to sample videos in a higher resolution. Solid lines depict required steps whereas dashed lines depict optional steps.

With *DGen*, we focus on a subset of environments: man-made environments. We identified a variety of examples for which *DGen* could be beneficial during data acquisition, e.g., instance detection within grocery stores, classification of elements within a warehouse, classification of static objects under illumination variations or even deformed objects over time with viewpoint variations. The overall idea is to use additional 3D data to annotate frames automatically. With only a few 3D annotations during acquisition, we automatically generate bounding boxes for every frame of a recorded video sequence. For simplicity, *DGen* supports the use of an additional camera mounted to the HoloLens to acquire images in a higher resolution, e.g., to detect smaller objects Finally, *DGen* also supports offline annotation refinements, e.g., for fine-grained product recognition in a retail setting. We base our approach on off-the-shelf hardware in order to reduce the hurdle to set up a similar system.

Figure 2 depicts the proposed workflow at a glance. We distinguish five different Steps: *Step 1)* represents an offline calibration step to attach an additional camera to the system. In *Step 2)*, we annotate objects with minimal 3D annotations, e.g., a shelf within a retail setting. Afterward, in *Step 3)*, we sample a video of the objects while moving the camera or changing the scene. In *Step 4)*, we refine the 3D annotati-

ons offline, e.g., to identify items within a shelf. Finally, we project the 3D annotations onto every frame of the video sequence in *Step 5)* and sample representative images and annotations. In the following, we describe every step in detail.

2.1 Calibrate System

Calibrating the proposed system is optional. It is only required if an additional camera is attached to the HoloLens. It is only necessary to determine the fixed relative translation and rotation between both.

To estimate the relationship between these two cameras, we determine the intrinsic parameters of every camera. We follow the procedure for single-view calibration as described in (Zhang, 2000). Afterward, we estimate the extrinsic parameters using the fundamental matrix (Faugeras et al., 1992; Hartley et al., 1992), i.e., the relative transformation from one camera to the other. We minimize the total re-projection error using pair-wise correspondences. As a result of proper calibration, we can project points from the HoloLens' space into the second camera's image space. In this paper, we use a Logitech Brio[5] to record 4K video sequences.

2.2 Annotate 3D Primitives

This step is a prerequisite for generating 2D bounding box annotations. The idea is to label objects of interest manually in 3D space. We label objects on sight using the built-in abilities of Microsoft's HoloLens. We use the user's gaze to annotate images and project a ray through the virtual center of the user's field of view and calculate the intersection with the continuously mapped 3D environment. On user input, we generate a new 3D annotation point. In later steps, we project these annotations into 2D image space to compute their bounding box.

We label objects of interest using 3D shape primitives, e.g., planes, cubes, or pyramids. They provide the usability which is necessary to speed up the overall annotation process and are rather efficient due to their simplicity. The 3D shape primitives are used to acquire bounding boxes in later steps. Next, we sample different video sequences of objects of interest.

2.3 Record Video Sequences

Data-driven approaches require multiple samples of a particular visual concept to understand it reliably. In this step, we sample one or more video sequences of physical objects. We sample multiple sequences since data-driven approaches shall be invariant to varying image conditions, e.g., viewpoint or illumination changes. Typically, it is required to include these varying conditions in the training data to achieve invariance.

We manually cause different internal and external variations while recording. Depending on the particular industrial use case, we create different conditions, such as changing viewpoint or illumination, occluding or deforming the object, or varying the environment to induce background clutter.

Manually inducing these variations allows us to sample adequate images of a particular object. We record the video sequence, 3D annotations and the position and orientation of the HoloLens to project the previously annotated 3D shape primitives into image space in every frame during the following steps.

2.4 Refine 3D Annotations

Refining the 3D annotations is optional. We use this step for fine-grained recognition tasks, e.g., when 3D annotated objects are composed of smaller objects.

In this step, we label refining objects within previous annotated 3D primitives. We annotate these refinements offline. We project rays from 2D image space onto the 3D shape primitives and calculate their 3D intersection. This allows us to sample new bounding boxes and brings two benefits:

- **One-shot Annotations**: we manually annotate a single image and transfer the annotations to the subsequent frames, and
- **Semi-automatic Annotations**: we use simple classifiers to detect objects and transfer the found annotations to subsequent frames.

Using the *one-shot annotation* function of *DGen*, only a single definition of a bounding box is required on one particular image to generate subsequent annotations. Through the ray cast of the 2D user annotations in an image onto the 3D shape primitive, we calculate the intersection of the ray and the shape primitive. Using the recorded camera positions and orientations for subsequent frames, we reproject these new annotations back onto the image plane in later steps. One-shot annotations decrease the annotation time for sequences dramatically.

In an industrial context, we can also *semi-automatically annotate* objects within previously on-sight annotated 3D shape primitives. The overall idea is to use reference images, e.g., sampled from the web, and a weak classifier, such as SIFT (Lowe, 2004), to detect correspondences. Using an approach similar as proposed in (Filax et al., 2017), we

[5]https://logitech.com/product/brio

detect different proposals. Further, we can incorporate a voting mechanism to determine valid detections and transform the found 2D detections onto the previously on-sight annotated 3D shape primitives. Using the found 3D locations and the recorded camera trajectory, we sample new detections, e.g., from challenging viewpoints, in which SIFT-based approaches typically fail (Yu and Morel, 2009).

2.5 Extract Bounding Box Annotations

In this step, we extract bounding box annotations for every frame in a video. We thereby project the 3D primitives and their refinements using the camera's trajectory. We forward project every vertex into image space (Hartley et al., 1992). Finally, we compute the bounding box of the projected points to acquire an annotation.

Afterward, we automatically sample multiple annotated images of a particular object instance. We have to determine whether to export a given frame or not to prevent a bias within the data. We envision different metrics, e.g., viewpoint change, illumination change, blur, deformation, or time. The metric, however, is up to the desired application. Thus, we sample every annotated frame of a given video in this paper.

3 THE MAGDEBURG GROCERIES DATASET

We recorded a dataset in a commonly known man-made environment: grocery stores. The *Magdeburg Groceries* dataset is online available at https://bitbucket.org/cse_admin/md_groceries. Our motivation for this dataset is two folded: demonstrate the usability of *DGen* and provide a benchmark to compare object recognition techniques in an industrial setting. The dataset consists of two parts: i) categorized training images of groceries in a studio setting and ii) annotated frames of real-world shelves from different stores.

As training images, we automatically collect the set of items from the web. Figure 3 displays exemplary training images from the dataset. All items have a resolution of 220x220 pixels. In total, we provide 23,360 images as annotated web links. We organized them in categories with a populated semantic hierarchy to reflect real-world product categories in typical grocery stores. Categories are linked with "is-a" relations. We provide 942 categories in total with 24.8 items on average.

As validation images, we recorded 48 video sequences in three different grocery stores with 953 frames

Figure 3: Sample training images from the *Magdeburg Groceries* dataset. These samples were collected in ideal studio conditions from real.de.

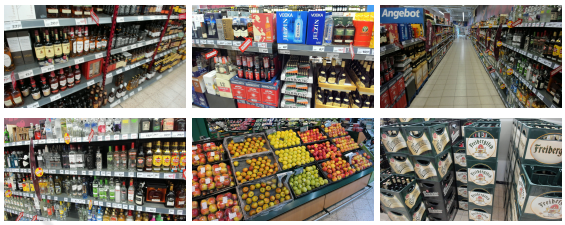


Figure 4: Example validation images from the *Magdeburg Groceries* dataset. These images are extracted from different videos. We did not enforce any viewpoint constraints, so these images suffer from blur, occlusion, and illumination changes.

on average. We spent approximately four hours per store. We used *DGen* to apply the procedure as described in section 2. We attached a Logitech Brio 4K to the HoloLens and calibrated the system (cf. sect. 2.1). Every video sequence was sampled in a resolution of 3840x2160 pixels. Examples are shown in figure 4. For every sequence, we annotated 1.7 shelves on average (cf. step 2.2) as a 3D bounded plane. In total, we annotated 83 shelves. Afterward, we recorded video sequences (cf. step 2.3). During the complete sequence, we sampled the position of the shelves, the trajectory of the HoloLens, and the sequence itself. We aimed at mimicking natural behavior and did not restrict viewpoint or illumination. Our dataset comprises 41,955 images annotated with challenging viewpoints and illumination changes.

Through *DGen*, we gained the possibility to annotate sequences offline with even more fine-grained annotations what results in more complex annotations as possible with state-of-the-art tools. To justify this claim, we briefly evaluate the effort required to annotate a shelf. We annotate a *single*, randomly selected shelf using the tool LabelImg. We created 57 bounding boxes with numeric identifiers in roughly 16 minutes. We were not able to identify more products due to the challenging viewpoint. The result is shown in

(a) 57 bounding boxes annotated with LabelImg.

(b) 76 bounding boxes annotated using *DGen*.

Figure 5: Comparing the proposed tool and LabelImg. We were able to annotate more items *DGen*.

Figure 6: Example fine-grained annotations from the *Magdeburg Groceries* dataset. On average we annotated 59 items on every frame.

figure 5. For comparison, we labeled the same shelf using *DGen*. The annotation of the same 57 bounding boxes requires 20 minutes. This is due to the more elaborate user interface of LabelImg. The shelf, however, comprises more items. Using *DGen*, we labeled 76 items in total. This was only possible due to the proposed approach from section 2.4. We were able to transfer our annotations from one frame to another and could identify more items. This results in more complex annotations, which are shown in figure 5.

Our dataset comprises 1,523 fine-grained item annotations in ten sequences in 17 shelves. Extracting these annotations generates 755,309 bounding boxes for 12,768 images in total. Figure 6 depicts examples. Annotations were done in roughly nine hours in total. Using the average time required to annotate a single item using a state-of-the-art tool, we estimate that it requires approximately 4,400 hours to label the same amount of data. We conclude that *DGen* gives a significant benefit during the fine-grained annotation of video sequences.

4 RELATED WORK

In this section, we summarize different related works. We give an overview of other state-of-the-art labeling tools and compare our dataset with others.

4.1 Fine-Grained Labeling Tools

We summarize the workflow of annotation tools in this section. We do not have space to review them all here but give a brief summary of some popular tools. LabelImg[6] is a popular open-source annotation tool. The workflow of LabelImg is to annotate multiple bounding boxes on a single image. LabelImg has some drawbacks when it comes to labeling sequences. Previously done annotations are not automatically transferred to the next image. This dramatically increases the annotation time for sequences. Other popular bounding box annotation tools have similar drawbacks, e.g., BBox Label Tool[7] or LabelBox[8].

In order to reduce the absolute time to acquire a large annotated database, it is typical to rely on parallelization (Russell et al., 2008; Deng et al., 2009). ImageNet (Deng et al., 2009) for example was built using thousands of workers. The task of annotating multiple bounding boxes on a single image remained unchanged but was dramatically parallelized. Although the approach reduces the absolute time, it does not influence the total time. The quality of annotations might vary as well, especially in details. Examples of this problem are subsequent frames. Because different persons annotate images, bounding boxes might wiggle in subsequent frames. Some annotations could be wrong because of deeply nested classes. These workflows typically incorporate some annotation verification step to increase the overall quality of annotations. This, however, requires additional effort. These observations motivated us to propose *DGen*.

4.2 Fine-Grained Product Datasets

In this section, we comprise different datasets for fine-grained product recognition. We list their properties in table 1. These datasets were collected manually, which requires a vast amount of work.

One of the largest collections for grocery data is the *openfoodfacts* (*off*) project. It aims at gathering information about products, i.e., product characteristics, ingredients, or nutrition facts. This dataset comprises over 560,000 product entries labeled with categories similar to ours. However, the data is submitted by different contributors and not automatically verified. Instead, the contributors shall detect errors. The user-provided images vary in size and quality. We evaluated random samples from the dataset and found that most products are present with up to three images per

[6] github.com/tzutalin/labelImg

[7] github.com/puzzledqs/BBox-Label-Tool

[8] labelbox.com

Table 1: Different properties of various datasets for grocery item recognition from the literature. All images in these datasets were annotated manually. The last row describes our dataset, generated using *DGen*. Note that the amount of annotated scene images is superior to any other dataset.

	Items			**Scenes**		
	Items	Images per Item	Categories	Images	Stores	Annotations
off	560,000	0-3	19,923	-	-	-
Freiburg Groceries	-	-	25	74	1	Presence
Grocery Products	3235	1	96	680	5	Groups
Grozi-120	120	2-14	-	11,194	1	Items
Grocery	10	370	1	354	40	Items
WebMarket	100	2-3	-	3153	1	Presence
Magdeburg Groceries Dataset	23,360	1	942	41,955 12,768	3 1	Shelves Items

product. An item typically has one image which displays the product, while others display the ingredient list or nutrition table. The dataset does not provide any scene images of the products on shelves.

The *Freiburg Groceries* dataset (Jund et al., 2016) comprises 4947 images of grocery products organized in 25 general categories. Images typically contain multiple product instances of a particular class. Jund et. al. collected almost 200 images per class on average. Most images show multiple products that do belong to the same general category. This dataset comprises 74 scene images of a single shelf. 36 distinct arrangements were taken in a controlled lab environment. An image contains different products which are partially occluded by others. The authors do not provide bounding boxes. Instead, they labeled the presence of particular classes.

The *Grocery Products* dataset (George and Floerkemeier, 2014) comprises 3235 images, spanning 98 hierarchically nested categories, downloaded from the web in studio conditions. Additionally, this dataset comprises 680 real-world scene images. These were taken in five different stores. Scene images are annotated as groups, and not as single product instances.

The *Grozi-120* dataset (Merler et al., 2007) comprises 2-14 images for 120 different products. The products were not categorized in contrast to the other datasets. The dataset comprises 676 training images collected from the web. The authors collected around six different images from different viewpoints for every item. Further, they extracted 11,194 scene images from 29 video sequence, which were taken with an off-the-shelf camera.

The *Grocery* dataset comprises ten different brands of tobacco packages (Varol and Kuzu, 2014) They collected 3701 training images with varying illumination and viewpoint changes. The authors raised 354 shelf images as validation images from roughly 40 different stores. Every instance of a tobacco package, which is present in the training data, was manually annotated. The authors reported that it took about one and a half minutes to annotate one scene image. This means annotating all scene images required almost nine hours.

The *WebMarket* dataset comprises over 3000 images (Zhang et al., 2007) split into two groups. Training images depict a single item, and validation images depict shelves. All images were collected directly within a single store. The authors collect 2-3 training images for 100 items which were placed on the floor. These images were taken with small viewpoint variations but no illumination changes. The authors collected 3153 scene images of different shelves from a single store using different cameras as validation data. Scene images were taken from various viewpoints and were annotated with product IDs. The authors do not provide bounding box annotations.

All of the described datasets required substantial manual annotations, either provided by a large community or by laborious data scientists. In contrast to those datasets, our dataset was almost completely generated with minimal user input. Item images were found on the web and scene images generated using *DGen*.

We visualized different properties of the datasets in table 1. The *off* dataset holds more item images than any other database because the *off* dataset is fed with by contributors from all around the world. However, it does not provide any scene images. We tried to combine this dataset with data generated by *DGen* but found that there is not enough overlap. The *off* dataset contains 65 products that are labeled to be sold in stores where we recorded shelves. Thus, we semi-automatically crawled images from the web.

With the proposed system we generated more annotated images as in any other dataset because we reproject annotations to subsequent frames (cf. sect. 2). We minimize the required manual work dramatically: We needed roughly two and a half minutes per video to annotate shelves. Including data acquisition, this

adds up to a total of 18 hours. During this period, we acquired 41,955 annotated images of shelves.

With an additional nine hours manual annotation refinement, we were able to acquire 755,309 bounding boxes in 12,768 images. In total, we annotated 871 unique products in ten video sequences.

5 EXPERIMENTS

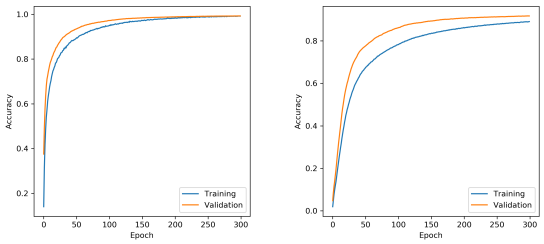
In this section, we conduct two different experiments with the *Magdeburg Groceries* dataset to discuss its usability for recognition tasks. First, we conduct a general recognition experiment, in which we classify a given image according to the flattened product category. Secondly, we conduct a fine-grained recognition experiment, in which we classify item crops provided by our dataset. Both experiments were conducted with the real-world scene images on a single NVIDIA GTX 1070 GPU. We used the TensorFlow implementation of the VGG-16 (Simonyan and Zisserman, 2015) network with ImageNet (Deng et al., 2009) weights.

5.1 Recognition of Grocery Shelves

To classify grocery shelves in a general recognition setting, we classify crops of the real-world scene images with 41,955 images in total. We use 69,929 crops, whereas 70% are used as training data and 30% are used for validation. We use a variant of the VGG-16 (Simonyan and Zisserman, 2015) network for fine tuning the fully connected layers. According to (Simonyan and Zisserman, 2015), we resize the cropped shelves to a fixed size of 224x224px which serve as input data. During training, the first 13 convolutional layers are initialized using ImageNet (Deng et al., 2009) weights and remain unchanged during the complete training procedure. We change the size of fc6 from 4096 to 64 and remove the fully connected layer fc7. Thus, we feed fc6 into the final classification layer (called fc8 in (Simonyan and Zisserman, 2015)) to classify the 37 different flattened classes. Further, we add dropout layers in between fc6 and fc8 with 30% dropout.

5.2 Fine-Grained Recognition of Grocery Items

For the second experiment, we classify cropped items of the *Magdeburg Groceries* dataset according to their unique 871 classes. Again, we use the complete set of real-world scene images with 41,955 images in total. During this experiment, we use a subset of 490,843 cropped grocery images with a size of 64x64 pixels or more. Similar to the general recognition experiment, we choose to use 70% of the cropped items as training data and 30% as validation data. Cropped items are resized to a fixed size of 64x64 pixels. Again, we use a variant of the VGG-16 (Simonyan and Zisserman, 2015) network for fine tuning the fully connected layers. The convolutional layers are initialized using ImageNet (Deng et al., 2009) weights and fixed during the training procedure. We change the size of fc6 from 4096 to 1024 and remove the fully connected layer fc7. Again, we use a dropout of 30% between the fully connected layers. Finally, we change the size of fc8 from 1000 to 871.



(a) Accuracy over time for a general recognition task. We were able to classify shelves with an accuracy of 99.28%.

(b) Accuracy over time for a fine-grained recognition task. We were able to classify grocery items with an accuracy of 91.83%.

Figure 7: Training history for both experiments. Both models converged within 300 epochs.

5.3 Quantitative Results

In both experiments, we minimized the categorical cross-entropy error using stochastic gradient descent with a batch size of 256 with an initial learning rate of 10^{-4} and a weight decay of 10^{-17}. Both models were trained for 300 epochs on the *Magdeburg Groceries* dataset. Similar to the test protocol for classification problems, we measured the performance of the learned model regarding accuracy. Figure 7 summarizes the results over time. The models converge smoothly during 300 epochs in both experiments. For the general recognition experiment on grocery shelves, we achieve an accuracy of 99.28% on the validation set. For the second experiment on fine-grained grocery items, we report an accuracy of 91.83%.

Both experiments demonstrate the usability of real-world scene images in the *Magdeburg Groceries* dataset. This is mainly because both we were able to use state-of-the-art classification models to distinguish general and fine-grained groceries. We were able to achieve comparable results with ease as we did not use extensive hyperparameter optimization me-

chanisms. This summarizes the quality of the proposed *DGen* tool. With the proposed tool we were able to acquire valuable training data fast and efficient. We minimized the effort to acquire the data dramatically as shown in section 4 while preserving the quality of the final dataset as shown in this section.

6 CONCLUSION

In this paper, we tackled the problem of migrating learning approaches to the industrial domain. Data-driven approaches like convolutional neuronal networks typically rely on images to learn visual properties of objects. A lot of annotated data is required to distinguish a large number of objects reliably. Therefore, data scientists have to annotate a large number of images laboriously.

Providing a dataset of 60,000 or more annotated images requires an enormous amount of laborious work. To overcome this, we proposed a system called *DGen* to generate bounding boxes in man-made environments with minimal user input. The core idea is to rely on the 3D trajectory of the camera while recording videos of objects of interest. We acquire the 3D trajectory and the user's input with Microsoft HoloLens's built-in sensors. Our tool additionally provides the opportunity to refine 3D shape primitive on-sight annotations for fine-grained recognition tasks. Finally, we project the on-sight 3D annotations and offline refinements onto frames of the recorded videos.

We have shown that *DGen* is well suited to generate fine-grained annotations via an application from the retail domain, in section 3. We described in section 4 that we were able to generate more annotated images than available in any other dataset to the best of our knowledge. We illustrated that it took only a fraction of the manual input reported in the literature and reduced the overall time to acquire annotated data for image recognition tasks. We conclude that *DGen* is especially well suited for industrial domains. In section 5, we demonstrated the applicability of the acquired dataset. We were able to recognize a sufficient number of objects in general and fine-grained grocery product recognition settings. In the future, we plan to extend our tool and provide additional datasets.

REFERENCES

Deng, J., Dong, W., Socher, R., Li, L.-J., Kai Li, and Li Fei-Fei (2009). ImageNet: A large-scale hierarchical image database. In *CVPR*, pages 248–255.

Faugeras, O. D., Luong, Q. T., and Maybank, S. J. (1992). Camera self-calibration: Theory and experiments. In *ECCV*, pages 321–334.

Fei-Fei, L., Fergus, R., and Perona, P. (2006). One-shot learning of object categories. *TPAMI*, 28(4):594–611.

Filax, M., Gonschorek, T., and Ortmeier, F. (2017). QuadSIFT : Unwrapping Planar Quadrilaterals to Enhance Feature Matching. *WSCG*.

George, M. and Floerkemeier, C. (2014). Recognizing products: A per-exemplar multi-label image classification approach. In *ECCV*, pages 440–455.

Hartley, R., Gupta, R., and Chang, T. (1992). Stereo from uncalibrated cameras. In *CVPR*, pages 761–764.

Jund, P., Abdo, N., Eitel, A., and Burgard, W. (2016). The Freiburg Groceries Dataset.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *NIPS*, pages 1–9.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In *ECCV*, pages 740–755.

Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 60(2):91–110.

Merler, M., Galleguillos, C., and Belongie, S. (2007). Recognizing groceries in situ using in vitro training data. *CVPR*, pages 1–8.

Netzer, Y. and Wang, T. (2011). Reading digits in natural images with unsupervised feature learning. *NIPS*, pages 1–9.

Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *TKDE*, 22(10):1345–1359.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2015). You Only Look Once: Unified, Real-Time Object Detection. *CVPR*, pages 779–788.

Russell, B. C., Torralba, A., Murphy, K. P., Freeman, W. T., Russell, B. C., Torralba, ·. A., Freeman, ·. W. T., Torralba, A., Freeman, W. T., and Murphy, K. P. (2008). LabelMe: A Database and Web-Based Tool for Image Annotation. *IJCV*, 77:157–173.

Simonyan, K. and Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition.

Varol, G. and Kuzu, R. S. R. S. (2014). Toward Retail Product Recognition on Grocery Shelves. *ICGIP*, 9443(2014):1–22.

Yu, G. and Morel, J. (2009). A fully affine invariant image comparison method. *ICASSP*, 26(1):1597–1600.

Zhang, Y., Wang, L., Hartley, R., and Li, H. (2007). Where's the Weet-Bix? In *ACCV*, pages 800–810.

Zhang, Z. (2000). A flexible new technique for camera calibration. *TPAMI*, 22(11):1330–1334.

Zhou, Z.-H. (2017). A Brief Introduction to Weakly Supervised Learning. *Natl. Sci. Rev.*, 5:44–53.