image2mass: Estimating the Mass of an Object from Its Image

Trevor Standley
Department of Computer Science
Stanford University
tstand@cs.stanford.edu

Ozan Sener
Department of Computer Science
Stanford University
ozansener@cs.stanford.edu

Dawn Chen
Google
Mountain View, CA
sdawnchen@gmail.com

Silvio Savarese
Department of Computer Science
Stanford University
ssilvio@stanford.edu

Abstract: Successful robotic manipulation of real-world objects requires an understanding of the physical properties of these objects. We propose a model for estimating one such physical property, mass, from an object's image. We collect a large dataset of online product information containing images, sizes, and weights. We compare several baseline models for the image-to-mass problem that were trained on this dataset. We also characterize human performance on the problem. Finally, we present a model that takes into account an estimate of the 3D shape of the object. This model performs significantly better than these baselines and compares favorably to the performance of humans. All models are tested on a held-out set of product data, as well as a relatively small dataset that we captured with a scale and a digital camera.

Keywords: Perception, physics, learning

1 Introduction

Imagine a robot that needs to hammer a nail. A desirable planner would first use its semantic knowledge of the task to look for a hammer in the scene. If no hammer is found, the robot would use its geometric and physics knowledge to find a heavy object that it can stably grasp, and then use that object for hammering the nail. Hence, successful planning and execution of physical tasks that a robot might perform requires a multimodal understanding of objects, including their semantic, geometric, and physical properties.

It is commonly believed that animals acquire a physical understanding of objects through continuous interaction with the physical world. Thus, researchers have taken the data-driven approach of letting robots interact with the world, which has recently been shown to be successful for learning to grasp [1, 2] and poke [3] objects. Although the results are promising, scaling this approach is a major issue. Even after months of experiments using multiple robots, the amount of physical data collected is nowhere near the amount of semantic [4] or geometric [5] data easily available.

Mass is one physical property of objects that critically influences optimal robot policies. Many robots have multiple end effectors, each of which is most effective for a different range of weights, and choosing the correct one for a particular object is of utmost importance [6]. Moreover, objects with different weights often require different joint configurations to manipulate most effectively. Although a robot might learn the mass of an object through interacting with it, being able to estimate the object's mass before any physical manipulation takes place saves time and prevents mistakes.

Furthermore, having accurate mass information for objects is important for physics and robotics simulations, which are often used for data augmentation in robot motion planning domains, e.g., [7, 8]. However, mass information is not easily available in such domains. Hence, a method for

1st Conference on Robot Learning (CoRL 2017), Mountain View, United States.

accurate estimation of an object's mass would be useful for both analytical and simulator-based data-driven approaches to robot motion planning. Finally, mass estimation has applications beyond robotics, including in agriculture, shipping, and medicine.

With these considerations, we tackle the problem of estimating the mass (weight) of an object from its image (image2mass). However, images alone are not sufficient, as an image of a toy car could be indistinguishable from an image of a real car. Therefore, we also give our models the dimensions of an object's bounding box (length, width, and height), which can be easily obtained either using an RGB-D camera or through metric calibration [9] of an RGB camera. Even with this size information, the problem of predicting mass is ill-posed and cannot be solved exactly. For example, empty milk cartons look the same as full ones. Nevertheless, we show that high-quality estimation is achievable by both machines and humans.

Unfortunately, this problem does not reduce to any better-studied problem, such as classification. Different instances of the same class can have vastly different masses (e.g., a chair can be made out of plastic or solid metal) and the vast number of categories exceeds the classification capabilities of current state-of-the-art systems.

One nice feature of the image2mass problem is the amount of data that is available. Many online stores such as Amazon.com include the physical dimensions, mass, and photos for each product. We obtain the product information for $\sim$3 million Amazon.com products, each with its weight, dimensions, and one image. After extensive cleaning, we produce a dataset with $\sim$150k products that have highly accurate mass and size information, covering a multitude of object classes. Taking dataset bias[10] into account, we also manually collect a smaller dataset by weighing, photographing, and measuring the dimensions of 56 household objects.

Using these datasets, we evaluate various sensible baselines using state-of-the-art deep learning architectures. Moreover, we conduct a study to characterize human performance on the image2mass task. Finally, we propose a novel model that combines geometry and density information, resulting in a performance superior to all the baselines by a large margin.

In summary, our contributions are:

- introducing the problem of mass estimation from an image in a class-agnostic way,
- producing a large dataset for the problem,
- a novel neural network architecture for this problem that combines geometry and physics, and
- extensive evaluation, including comparison with several baselines and human performance.

2 Related Work

To the best of our knowledge, there is no prior work addressing this problem for generic objects in a class-agnostic way. In a few lines of work, weights are estimated for a narrow domain like Alaskan salmon [11], beef [12, 13], pigs [14], and human body parts [15]. The authors of [16] designed a system for computing the mass of a small number of food classes such as bananas or bread. The system estimates the volume of an object using two images that are each from a different viewpoint. It then classifies the object and looks up a pre-measured density for that object class as an estimate of the object's density.

Our method is also similar to [17], which uses videos of simple Platonic solids with an object tracker and a physical simulator to learn parameters of the simulator, including the object's mass. Our method does not need a simulator, object tracker, or videos; instead it uses 2D images and generalizes beyond Platonic solids to real-life objects.

Other lines of work attempt to estimate physical properties such as shape [18, 19, 20], sound [21], and materials[22, 23]. We see our work as complementary to these.

3 Image2mass Dataset

As far as we know, there is no existing large-scale dataset containing both objects' images and their mass information that is easily available. However, large amounts of such data can be found on online stores, which often list photos of products along with their weight and size information. The

challenges are to find the objects that are most useful for the image2mass task and to extract the relevant information from their product pages.

3.1 Large-Scale Amazon.com Data

Using the Amazon Marketplace Web Service (MWS) API, we collected the images of $\sim$3 million products, together with their listed weights and dimensions.

The MWS API returns information for the top ten results of a natural language product search query. Combining a large collection of physical nouns with a large collection of adjectives taken from WordNet [24], we produced and ran a large number of product search queries, such as "ceramic ultraviolet lamp." For every distinct result that the API returned, we verified that a product image was available and that the product information listed the object's weight and dimensions before adding the product to the dataset. The average query resulted in 0.724 data items collected. The API's rate limits required data collection to be performed continuously over the course of a year.

Amazon product images are typically 500×500 pixels and are easy to segment because each object is typically on a perfectly white background. Importantly, Amazon sells a large set of existing object types. However, because there are systematic problems in the data, extensive filtering is required. Here are some reasons that the data for a particular product might be problematic:

- The number of objects pictured does not match the number sold.
- The product is shown in its box.
- The product is not easy to segment from its background.
- The weight or measurements are inaccurate.
- The listed weight or measurements do not match what is pictured in the image.

In order to discard such data items, we developed a multi-stage filtering pipeline. The first ten stages of this pipeline use heuristics (e.g., discard products listed as containing multiple objects) to filter out about 90% of items that have one of the above problems. The final stage of the pipeline uses a binary classifier to remove any item that recognizably falls under one of the above categories but wasn't removed by the prior stages.

Although the filtering pipeline removes a lot of possibly useful data, it still results in $\sim$150k images. For all experiments, this was divided into $\sim$141k training examples and $\sim$6k validation examples.

Amazon Test Set: In order to produce the highest-quality dataset to use as one of our test sets, we chose a random subset of the 150k images and manually removed problematic samples. This filtering was done prior to performing any experiments in order to keep the evaluation unbiased. This process resulted in 924 test samples from the initial subset.

3.2 Household Test Set

In order to ensure that our models generalize to real-world objects, we separately collected a dataset of 423 images of 56 household objects. These objects were photographed on a white sheet using a compact digital camera on a cloudy day outdoors. Each object was photographed at an average of 7.5 different viewpoints. The objects were manually weighed on an appropriate scale and their dimensions were measured. The objects' images were segmented using the GrabCut algorithm [25] and placed on pure white backgrounds. One example image from each object can be seen in Figure 1.

4 Shape-Aware Model

Here we describe our deep architecture for estimating the mass of an object using an RGB image and its dimensions. Figure 3 provides an overview of our model.

An object's mass is the product of its density and its volume. Density is determined by the object's materials, whereas volume is determined by the object's 3D geometry. Our main idea is to capture both density and volume information, and to simply multiply them to estimate an object's mass.

Density and volume are each captured by a different neural network tower. The *density tower* is CNN-based and the smaller *volume tower* is fully connected. In addition to these two towers, our model has a module for estimating an object's 3D shape. This *geometry module* is trained before

Figure 1: A single image for each of the 56 household test items.

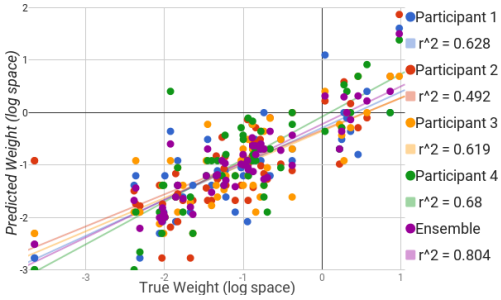

Figure 2: Human correlation with ground truth (log space).

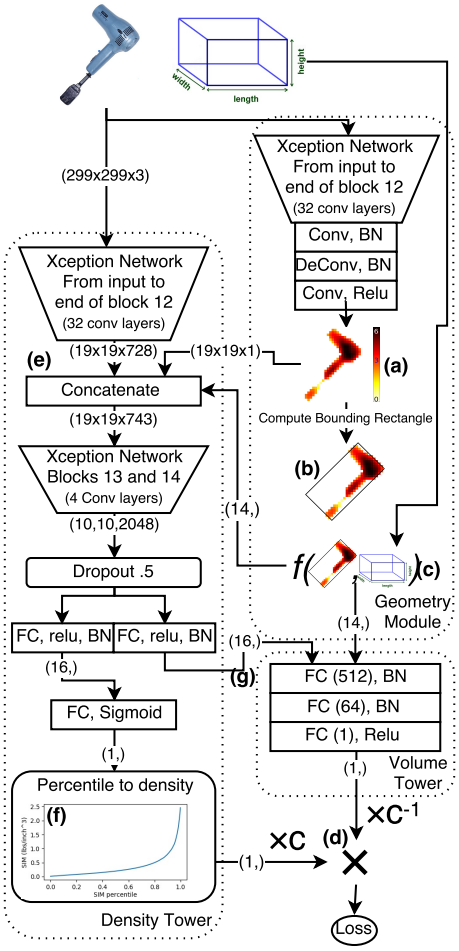

Figure 3: Our shape-aware network diagram. Locations **(a)** through **(g)** are described in more detail where marked in the text.

the other two modules. Its output is processed into features that are fed into both towers, which are trained jointly. We explain the details of each component in the following subsections and illustrate them in Figure 3.

4.1 Geometry Module

The main function of the geometry module is to take an RGB image and produce an estimate of the thickness of the object at each pixel. It also computes a number of properties based not only on this estimated geometry, but also on the given object's physical dimensions.

In principle, we could obtain an estimate of an object's shape from an existing 3D shape predictor [18, 20]. Unfortunately, these models turned out not to suit our needs. Instead, we found that predicting a 2D thickness mask was sufficient to capture the relevant 3D shape of an object. A thickness mask is an estimate of the thickness of the object at each pixel in the image. It is distinct from a depth mask in that it captures hollowness and the extent of the back face.

The geometry module is trained separately from the other components of our model using ShapeNet [5] data. Using ShapeNet's 51k 3D models of objects, we created a training set of 500k images, each paired with its thickness mask. We used these data to fine-tune a modified Xception network [26] that was pre-trained on ImageNet. After training, the geometry module produces a 19×19 thickness mask **(a)**, which we use to compute a bounding rectangle **(b)**. The geometry

module uses these to compute a number of geometric properties that are designed to aid in volume understanding, which are fed into both the volume and density towers **(c)**.

Computing a volume estimate from the thickness mask is trickier than might appear at first glance. First, thickness masks exist in pixel space, but we need our volume in physical length units such as $inches^3$. Worse yet, dimensions are listed on Amazon with respect to a bounding box that is aligned with an unknown coordinate system.

We estimate this bounding box by computing a minimum-area bounding rectangle on the thickness mask. Figure 3 shows an example thickness mask with its bounding rectangle for an object from the household test set **(b)**. We assume that the depth dimension of the bounding box is the maximum value in the thickness mask. Since this assumption may not be true, and our bounding box's alignment may not be accurate, we allow the volume tower to learn the precise computation for an object's volume, and it can learn to deal with exceptions.

For this purpose, the geometry module computes the following 14 geometric properties **(c)**: the length (1) and width (2) of the bounding box; the maximum entry in the thickness mask (3) (which is treated as the depth of the box) the bounding rectangle's center (4,5); the bounding rectangle's angle (6); the sum of all values in the thickness mask (7); the number of non-zero entries in the thickness mask (8); the area of the bounding rectangle (9); the volume of the bounding box (10); the proportion of voxels in the bounding box that are occupied by the object, o_1 (11); the proportion of pixels in the bounding rectangle that are occupied o_2 (12); the 3D volume estimate (13) ($o_1 \times L \times W \times H$); and the 2D volume estimate (14) ($o_2 \times L \times W$).

4.2 Volume Tower

The volume tower consists of two fully connected layers and takes as input the set of 14 features computed by the geometry module. Additionally, we take 16 features **(g)** from a final-layer representation of the density tower so that the volume tower has access to whatever the CNN can encode about volume. The output of the volume tower is a scalar, which is multiplied by the output of the CNN density tower to compute the output of the model **(d)**.

4.3 Density Tower

The density tower takes as input an RGB image, a thickness mask, and the set of 14 geometric properties and estimates an object's density.

The tower is a modified Xception network. This architecture worked better empirically than VGG [27] and ResNet [28]. Midway through the network's computation **(e)**, it receives thickness mask information as well as the 14 geometric properties from the geometry module. The network employs a fully connected layer for producing a single scalar that captures density information.

The density tower uses two activation functions at the last layer to ensure that its output has approximately the same distribution as the densities of training set objects. The first function is a sigmoid that squashes the output to be approximately uniform in $(0, 1)$, which represents the percentile of an object's density. The second function **(f)** is a pre-computed, piece-wise linear activation function that converts the percentile into a density and is derived from the sorted list of the *scale-invariant masses* (SIMs) of objects in the training set. The scale-invariant mass of an object is a proxy for its true density, computed as $^{mass}/_{L \times W \times H}$. Because most real-world objects' densities fall within a certain range and only a few objects have very high densities (which is true for the training set as well), the second activation function has the effect of making very high densities rare.

4.4 Combining Tower Outputs

To correctly combine volume and density information into mass, multiplication **(d)** is necessary. However, multiplying the outputs of two neural networks is tricky. If the two outputs have dramatically different scales, no single learning rate can work for both networks. However, for any two multiplicands x_1 and x_2, $x_1 x_2 = (cx_1)(\frac{1}{c}x_2)$ for any c, so we can choose c such that $cx_1 \approx \frac{1}{c}x_2$ to resolve the issue of different scales. In our case, the volume tower's output typically dwarfs that of the density tower, so x_1 is the density tower's output and x_2 is the volume tower's output. We found that a c of 10 worked well, though quick tests suggest that a wide range of values near 10 also work. We used $c = 10$ during the entire training process and across different experiments.

4.5 Loss Function

We found the absolute log difference error (ALDE, introduced by [29]), $|\ln t - \ln p|$, to be a good loss function for the image2mass problem. This loss function can be thought of as something like the difference in the order of magnitude between the target and prediction. It has the nice property that it penalizes *relative* rather than *absolute* error (so that larger deviations are allowed for heavier objects than for lighter objects), unlike mean absolute error or mean squared error (which give greater importance to heavier objects by penalizing absolute error). Moreover, unlike mean absolute percentage error, $|\frac{t-p}{t}|$, ALDE does not lead to models that systematically produce underestimates [29].

5 Human Performance on the image2mass Task

Four human participants were trained to estimate the weights of objects by practicing on examples from the Amazon test set. After training, participants were asked to estimate the weights of the 56 items in the household test set given only their images and size dimensions. Participants were allowed to use a ruler to help them visualize the given dimensions and to hold a one-pound weight. The true weight of each item was provided at the end of each trial. Participants often wondered aloud about the various materials in an object while they tried to estimate its weight. Many of them enjoyed the task.

According to the Weber-Fechner law, the subjective intensity of a perceptual phenomenon is proportional to the log of the true stimulus intensity [30]. For example, the perceived difference between a 1 lb object and a 2 lb object is similar to that between a 10 lb object and a 20 lb object. Thus, we examined the correlation between participant weight predictions and ground-truth weight in log space, shown in Figure 2. All data points are shown for each participant, as well as the regression line and r^2. The performance of a hypothetical human ensemble, which makes the median participant weight prediction for each object, is also shown. Although not shown in the graph, participants were surprisingly well-calibrated (i.e., their regression lines in the original space had intercepts close to 0 and slopes close to 1). Moreover, the weights of most objects were not consistently over- or under-estimated by participants.

Ultimately, these results should be taken with a grain of salt. People rarely need to produce explicit weight estimates; such estimation is typically an unconscious part of routine physical manipulation of objects. It is not clear how well this study exposes people's intuitions about objects' weights.

6 Experiments

6.1 Baseline Models

***k*-Nearest Neighbors** Our first baseline uses the *k*-nearest neighbors method. We extract features from the Average Pooling layer of Xception trained on ImageNet. We train a *k*-nearest neighbor regressor to predict the mass of an object. We improved the *k*-NN baselines significantly by predicting the scale-invariant mass instead of the mass directly. At test time, we multiply the model's output by the object's dimensions to recover the mass. For both methods, the best k was 40.

Pure CNN Baseline Our best purely CNN-based model predicts the real-valued percentile of the object's SIM using only an image. We use our training set to fine-tune an Xception network in which we've added a dropout layer and a fully connected layer at the end of block 14. The scalar-valued result is passed through a sigmoid function before the loss is calculated, for which we use mean absolute error between the predicted percentile and the actual percentile. At test time, we convert the percentile into a SIM, and then multiply by the object's dimensions to recover the mass.

In order to independently quantify the importance of our loss function, we also report the results of a simple CNN like the above, but with the percentile-to-SIM activation function applied at training time using the mALDE (mean ALDE) loss. This baseline shows how the model behaves without the geometry module and with the volume module replaced by simply $L \times W \times H$.

Finally, as an ablation study, a model was trained without the geometry module. The volume tower was fed only length, width, height, and their product, along with features from the density tower.

6.2 Implementation Details

All models were trained on two Pascal Nvidia Titan X GPUs using Keras [31] with the TensorFlow [32] backend. Training was halted after 50 passes (epochs) through the data. Validation performance was calculated after each pass and the best model was used in testing. Training time averaged about two days for each model. No hyper-parameter search was performed and all models were optimized using Adadelta [33]. Thickness masks and geometric properties were pre-computed for the training set using the geometry module, which was trained ahead of time and frozen.

All models rely heavily on data augmentation. During training, images were randomly rotated up to 360°, translated by up to 12% in any direction, scaled to between 80% and 107% of their original size, and mirrored 50% of the time. We also employed random color-space shifts.

At test time, our model and the CNN baselines are fed eight images: the original image, its mirror image, and these images rotated by 90°, 180°, and 270°. The median result across these eight images was reported. If we use this method, we can make predictions for about 13 images per second.

6.3 Metrics

We report the performance of each model on several metrics (the true mass is denoted by t and the predicted mass is denoted by p):

- Absolute log difference error (ALDE): $|\ln t - \ln p|$, which is our training loss function.
- Absolute percentage error (APE): $|\frac{t-p}{t}|$, which is a simple metric, but prefers models that underestimate systematically.
- Coefficient of determination in log space: (r^2_{ls}), which is the squared correlation between $\ln t$ and $\ln p$. This metric is invariant to a model's calibration; additive and multiplicative constants are irrelevant. This is useful for comparing against uncalibrated predictors such as humans. Log space is used to correspond to human perception.
- Min ratio error (MnRE): $\min(\frac{p}{t}, \frac{t}{p})$, which we believe is the ideal metric. It has a value of 0 when the model always predicts zero or infinity and a value of 1 when the model makes perfect predictions, and is not biased toward models that over- or under-estimate. Moreover, the measure does not over-weight heavier instances because it looks at the ratio of true and predicted weights rather than their absolute difference, just as humans do. Finally, MnRE has an intuitive interpretation. For example, if the min ratio is 0.5, that means the estimate is typically off by a factor of 2.
- We also report the proportion of the time a model was within a factor of 2 of the true mass (which we denote by q).

6.4 Results

We computed each metric for all models on both the Amazon test set and the household test set. Tables 1 and 2 show the performance of all models. The results of our shape-aware model are superior to every baseline on both test sets. They are even competitive with human scores. Furthermore, the poor performance of both *k*-NN models supports our claim that semantically similar objects often have very different masses. Finally, we see robust agreement in the results between the two datasets, which suggests that neither dataset size nor dataset bias is much of a problem.

Because each object in our household test set was photographed from multiple viewpoints, multiple predictions for the same object can be averaged to create a multi-view prediction. In this setting, the performance of our model improves significantly, surpassing the median human score on mALDE, mAPE, and r^2_{ls}. An ensemble of humans, however, was far superior to any model, and serves to show what could be possible with only an image and size information.

A t-SNE [34] analysis showed that the CNN baselines readily encoded features related to an object's materials, but it was hard to find much evidence that the model was capturing shape or volume information from the supervision it was given. Furthermore, while the baseline models had access to size information at *test time*, they were not allowed to *learn* anything on the basis of size, unlike our two-tower shape-aware model.

Table 1: Model Performance on the Household Test Set (56 items, 423 images)

model	mALDE (↓)	mAPE (↓)	r^2_{ls} (↑)	**mMnRE** (↑)	q (↑)
Xception k-NN	1.454±0.094	6.490±1.155	0.079	0.341±0.026	0.270±0.043
Xception k-NN (SIM)	0.781±0.059	1.443±0.216	0.525	0.534±0.025	0.537±0.049
Pure CNN	0.680±0.056	0.811±0.117	0.570	0.575±0.023	0.660±0.046
CNN with ALDE	0.666±0.050	0.734±0.083	0.576	0.573±0.023	0.629±0.047
No Geometry	0.520±0.039	0.742±0.092	0.605	0.638±0.021	0.728±0.043
Shape-aware	**0.465**±0.040	**0.685**±0.088	**0.691**	**0.675**±0.022	**0.766**±0.041
Shape-aware multi-view	0.437±0.110	0.637±0.235	0.701	0.693±0.061	0.786±0.111
Human Median	0.459±0.132	0.583±0.522	0.628	0.675±0.062	0.798±0.169
Human Ensemble	0.323±0.082	0.398±0.146	0.804	0.754±0.050	0.877±0.088

Table 2: Model Performance on the Amazon Test Set (924 items)

model	mALDE (↓)	mAPE (↓)	r^2_{ls} (↑)	**mMnRE** (↑)	q (↑)
Xception k-NN	0.914±0.052	2.762±0.458	0.358	0.504±0.018	0.488±0.033
Xception k-NN (SIM)	0.677±0.034	1.141±0.108	0.720	0.570±0.016	0.584±0.032
Pure CNN	0.550±0.032	0.711±0.072	0.742	0.634±0.015	0.696±0.030
CNN with ALDE	0.592±0.034	0.849±0.103	0.713	0.616±0.016	0.672±0.031
No Geometry	0.490±0.027	**0.613**±0.062	0.751	0.657±0.014	0.746±0.029
Shape-aware	**0.470**±0.028	0.651±0.071	**0.767**	**0.672**±0.014	**0.767**±0.028

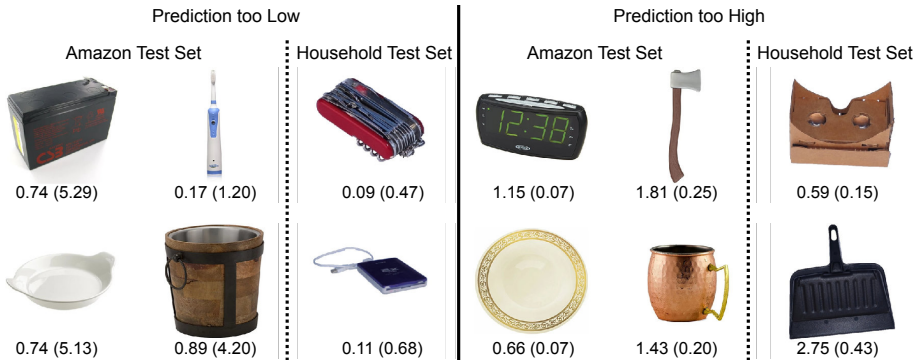

Figure 4: Failure cases of our model. True and predicted weights are reported as predicted (true).

Failure Cases: Figure 4 shows objects for which the model prediction is significantly different from the listed weight. Many of these errors seem to be due to a failure to identify the correct material. For example, the surprisingly light axe must be made out of plastic, and the white dish seems to be made out of plastic but is actually ceramic. In other cases, the geometry model failed to provide the proper thickness. The upside-down dust pan is only a few millimeters thick and the metal cup is actually very thin. Yet other errors reflect a lack of knowledge about real-world objects. For example, the car battery is heavier than predicted because it is filled with lead. The model predicted the Cardboard headset to be lighter than it actually was because Amazon doesn't sell many items made completely out of cardboard. Also common were errors in product listings. The nine-inch alarm clock cannot possibly weigh only .07 pounds (the weight of a quarter).

7 Conclusion

We introduced the problem of estimating the mass of an object from its image. We collected a large-scale dataset of objects' images, size dimensions, and weights from Amazon.com, as well as a high-quality dataset containing household objects that were manually photographed and weighed. We then described a neural network model composed of physically meaningful modules, including modules for computing volume and density information. We also examined human performance on the image2mass task.

Experimental results on the large-scale Amazon dataset as well as the household dataset validates our modular approach, showing significant improvement over CNN baselines and performance that is competitive with humans. We believe that our results will have many applications, including model predictive control and simulator design.

Acknowledgments

We acknowledge the support of Toyota (1191689-1-UDAWF) and Nvidia. Furthermore, we are grateful to our assiduous study participants, Chris Hall, Alexa Cohen, Cevin Freed, and Winona Krieger. We also thank Animesh Garg for insightful discussions about robots. Finally, thank our three anonymous reviewers, who offered excellent feedback.

References

[1] L. Pinto and A. Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *ICRA*, pages 3406–3413, 2016.

[2] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *IJRR*, 2016.

[3] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine. Learning to poke by poking: Experiential learning of intuitive physics. In *NIPS*, 2016.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009.

[5] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q.-X. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015.

[6] N. Correll, K. E. Bekris, D. Berenson, O. Brock, A. Causo, K. Hauser, K. Okada, A. Rodriguez, J. M. Romano, and P. R. Wurman. Lessons from the amazon picking challenge. *CoRR*, abs/1601.05484, 2016. URL http://dblp.uni-trier.de/db/journals/corr/corr1601.html#CorrellBBBCHORR16.

[7] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, and K. Goldberg. Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In *ICRA*, 2016.

[8] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*, 2017.

[9] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003. ISBN 0521540518.

[10] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *CVPR*, pages 1521–1528, 2011.

[11] M. O. Balaban, G. F. nal engr, M. G. Soriano, and E. G. Ruiz. Using image analysis to predict the weight of alaskan salmon of different species. *Journal of Food Science*, 75(3):E157–E162, 2010. ISSN 1750-3841.

[12] c. Taşdemir, M. Yakar, A. Ürkmez, and c. İnal. Determination of body measurements of a cow by image analysis. In *Proceedings of the 9th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing*, CompSysTech '08, pages 70:V.8–70:1, New York, NY, USA, 2008. ACM. ISBN 978-954-9641-52-3.

[13] Y. Bozkurt, S. Aktan, and S. Ozkaya. Body weight prediction using digital image analysis for slaughtered beef cattle. *Journal of Applied Animal Research*, 32(2):195–198, 2007.

[14] Y. Yang and G. Teng. Estimating pig weight from 2d images. In *CCTA*, volume 259 of *IFIP Advances in Information and Communication Technology*, pages 1471–1474. Springer, 2007.

[15] S. N. Le, M. K. Lee, and A. C. Fang. *Non-linear Image-Based Regression of Body Segment Parameters*, pages 2038–2042. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-540-92841-6.

[16] C. Chaithanya and S. Priya. Object weight estimation from 2d images. 10:7574–7578, 01 2015.

[17] J. Wu, I. Yildirim, J. J. Lim, B. Freeman, and J. Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *NIPS*, 2015.

[18] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[19] Q. Huang, H. Wang, and V. Koltun. Single-view reconstruction via joint analysis of image and shape collections. *ACM Trans. Graph.*, 34(4):87:1–87:10, July 2015. ISSN 0730-0301.

[20] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. *CoRR*, abs/1612.00603, 2016. URL http://arxiv.org/abs/1612.00603.

[21] A. Owens, P. Isola, J. H. McDermott, A. Torralba, E. H. Adelson, and W. T. Freeman. Visually indicated sounds. *CoRR*, abs/1512.08512, 2015.

[22] S. Bell, P. Upchurch, N. Snavely, and K. Bala. Material recognition in the wild with the materials in context database. *CoRR*, abs/1412.0623, 2014. URL http://arxiv.org/abs/1412.0623.

[23] L. Sharan, C. Liu, R. Rosenholtz, and E. H. Adelson. Recognizing materials using perceptually inspired features. *International Journal of Computer Vision*, 103(3):348–371, 2013. ISSN 1573-1405.

[24] G. A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, Nov. 1995. ISSN 0001-0782.

[25] C. Rother, V. Kolmogorov, and A. Blake. "grabcut": Interactive foreground extraction using iterated graph cuts. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 309–314, New York, NY, USA, 2004. ACM.

[26] F. Chollet. Xception: Deep learning with depthwise separable convolutions. *CoRR*, abs/1610.02357, 2016.

[27] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[28] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[29] C. Tofallis. A better measure of relative prediction accuracy for model selection and model estimation. *JORS*, 66(8):1352–1362, 2015.

[30] G. Fechner, H. Adler, D. Howes, and E. G. Boring. *Elements of psychophysics*. Number v. 1 in Elements of Psychophysics. Holt, Rinehart and Winston, 1966.

[31] F. Chollet et al. Keras. https://github.com/fchollet/keras, 2015.

[32] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL http://tensorflow.org/.

[33] M. D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

[34] L. van der Maaten and G. E. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.